Massachusetts Institute of Technology
The Artificial Intelligence Laboratory

A.I. Memo 755

January, 1984

THE COPYCAT PROJECT:

An Experiment in Nondeterminism and Creative Analogies

Douglas R. Hofstadter
Associate Professor
Computer Science Department
Indiana University

Currently: Visiting Scientist
Artificial Intelligence Laboratory
Massachusetts Institute of Technology

ABSTRACT

A micro-world is described, in which many analogies involving strikingly different concepts and levels of subtlety can be made. The question "What differentiates the good ones from the bad ones?" is discussed, and then the problem of how to implement a computational model of the human ability to come up with such analogies (and to have a sense for their quality) is considered. A key part of the proposed system, now under development, is its dependence on statistically emergent properties of stochastically interacting "codelets" (small pieces of ready-to-run code created by the system, and selected at random to run with probability proportional to heuristically assigned "urgencies"). Another key element is a network of linked concepts of varying levels of "semanticity", in which activation spreads and indirectly controls the urgencies of new codelets. There is pressure in the system toward maximizing the degree of "semanticity" or "intensionality" of descriptions of structures, but many such pressures, often conflicting, must interact with one another, and compromises must be made. The shifting of (1) perceived boundaries inside structures, (2) descriptive concepts chosen to apply to structures, and (3) features perceived as "salient" or not, is called "slippage". What can slip, and how, are emergent consequences of the interaction of (1) the temporary ("cytoplasmic") structures involved in the analogy with (2) the permanent ("Platonic") concepts and links in the conceptual proximity network, or "slippability network". The architecture of this system is postulated as a general architecture suitable for dealing not only with fluid analogies, but also with other types of abstract perception and categorization tasks, such as musical perception, scientific theorizing, Bongard problems and others.

This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the laboratory's artificial intelligence research is provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N00014-80-C-0505.

CONTENTS

GOALS OF COPYCAT

OVERVIEW

In the Copycat project, we are attempting to make a fluid model of the process of analogical thought. Note that we did not say "analogical reasoning". This is because we do not view cognition as carried out by a "reasoning engine" at whose beck and call there is an "analogy-making engine". We view the analogical facility as pervading cognition at every level -- in other words, as the driving force behind cognition. We view high-level cognitive functions (such as reasoning, planning, and use of natural language) as abilities that would evolve naturally from a sophisticated capacity to carry out analogical thinking, rather than the reverse. We believe that analogies are a blend of perception and memory retrieval, and that the best way to model doing them is to adapt techniques that work for perception -- especially vision and audition, sometimes considered to be merely peripheral aspects of mentality, but which we consider to be central -- a view well expressed by the incisive slogan "perception equals cognition".

In this project, we are focusing on analogies in an idealized domain. This domain was not chosen arbitrarily, but emerged as a product of years of honing down the question of "What makes a good analogy?", always searching for an ever smaller domain where the essence of analogy still emerged clearly. We feel that the end product of this search, our alphabetic micro-world, deserves to be thought of as more than a "toy domain". Rather, we like to think of it as an "ideal gas" of analogy-ology (the study of analogies), in that it is a domain that never arises in the real world, yet in a capsule form summarizes and isolates the major properties of analogies. It allows us to test theories about analogy-making in a pure, uncontaminated way. Another such "ideal gas", although more complex than ours, is the universe of "Bongard problems" (see **Bongard** and Chapter XIX of **Hofstadter:GEB**).

This report is organized in the following way. First we delineate what we consider to be the central qualities of analogies in the abstract. (This discussion is extended in APPENDIX 2.) Then we present Copycat's alphabetic micro-world (or meta-domain, as we sometimes call it). This includes "Platonic" entities (types), "ephemeral" entities (tokens), low-level (syntactic) links, high-level (semantic) constructs, and so on. Having presented our domain, we exhibit a series of analogy problems in it, designed to raise fundamental issues, and for us to try out both on people and on machines. This leads to a discussion of the sticky matter of "good" versus "bad" answers in analogy-making. Then we move on to our methodology, a section broken into two parts. The first part sketches the architecture of our already-developed system called "Jumbo", at whose core is a biologically inspired stochastic parallelism guided by urgencies, happinesses, and a notion of "computational temperature". (In APPENDIX 3, we compare Jumbo with recent work by several other groups interested in stochastic search techniques controlled by a temperature variable.) In the second part of the methodology section, we follow one subtle analogy problem from beginning to end as the Copycat system (as we now envision it) might actually treat it. We describe our theory of how probabilistically spreading activation in our concept network (the "Slipnet"), mediated by a Jumbo-like parallelism, can fluidly guide perception and lead to the discovery of subtle analogies.

ANALOGIES, ROLES, FRAMEWORKS, TRANSLATION

Briefly stated, the essence of our problem is this. Given two large frameworks or "worlds", each woven together in its own idiosyncratic way, and some highlighted fragment of one, where is the "same" fragment in the other? How can one export a role from one framework to another? Given two rival mappings both of which claim to establish such role correspondences, how can one determine

which is deeper? Is there any hope of finding general ways of answering this question, ways that transcend specific domains? (See **Moore & Newell.**)

Deep analogy problems are encountered in the act of translation between languages, just as in translation between the personal viewpoints of individuals. (See **Steiner.**) Indeed, we view translation between languages as a an excellent source of very complex but fascinating analogy problems, because there, two overarching frameworks are given: the two languages and their cultures. In one of these frameworks, attention has been drawn to a small area, to some special relationships and interconnections. Some piece of the vast framework has been singled out in a special light. It is the job of the translator to look inside the target framework and try to locate a similar small area, with some similar relationships and interconnections In short, what things play the same roles in the target framework as the things in the original framework? How can one characterize the roles of the original items abstractly enough so that there is hope for locating items in the new framework that fill the same roles? What things can be ignored, and what things constitute the essence?

Often a literal translation entirely misses what was being said by allusion or by connotation. This is because no two languages share the same *topology* and *topography* of their semantic spaces. By "topology", we mean what is near to what: in essence, a topology is a metric, or *proximity measure*, between concepts in the semantic space. By "topography", we mean what is important or salient: in essence, a topography is a *centrality function* for concepts in the semantic space. One can envision a language as having certain "nuclei" (words and stock phrases) surrounded by "clouds" (associations). These local features, taken together, weave the global net of the language, or its topology. The frequencies with which various routes are followed define the importance of the various nuclei, or in other words its topography.

These notions of the topology and topography of a semantic space arise in our model of analogy-making -- in particular, in our "Slipnet", or concept network (originally proposed in **Hofstadter:GEB**, Chapter XIX). The topology of our Slipnet is defined by certain concepts realized as nodes, and by the links that define proximity, or what we call "slippability" (which gives the Slipnet its name). The Slipnet's topography is defined by our attaching to each node a "semanticity", roughly defining its degree of abstractness. The reason for our equating abstractness with importance is our belief that the degree of depth and power of an analogy is proportional to the degree of its abstractness, and so we wish to bias our system towards framing its roles and analogies in the most abstract ways possible.

The act of deep translation (as contrasted with literal translation) involves recognizing which points in the semantic space are being highlighted by the input text, and then, instead of choosing the nearest points in that space allowed by the target language, choosing a set of points whose abstract relationships most closely resemble those embodied in the original text. However, there is a struggle here, since too much displacement can result in an overly metaphorical or abstract translation. Balance is needed. This is a critical focus in our research on analogies: how to reconcile "syntactic" and "semantic" pressures. By "syntactic pressures", we mean those that wish to emphasize more superficial qualities, whereas for us "semantic pressures" are those that wish to emphasize more abstract features at the expense of all others. This is a running battle in analogy. (See **Gentner** for discussions of "structure-mapping" and "systematicity" in analogy-making.) In short, one could say that our project views analogy as a highly generalized form of translation: translation between frameworks. The central problem is how to formulate a theory that will allow a machine to achieve a graceful compromise between forces pushing for literal translation and forces pushing for very abstract, metaphorical connections.

COPYCAT'S DOMAIN, CONCRETELY DESCRIBED

The concrete domain of Copycat is the alphabet, namely, the letters "A" through "Z". (Henceforth we will assume there is no need to be formal, and will not insert quote marks around letters and other structures in the domain. Context should make it perfectly obvious what is meant.) The letters are not to be considered as visual objects (shapes), but as abstract entities having a "Platonic" sequential order. In fact, we think of these timeless Platonic entities as letter *types*, as distinguished from letter *tokens*, which are ephemeral objects out of which temporary structures are built. There are thus relationships that hold, timelessly, among the Platonic letter types and higher Platonic abstractions, and then there are other relationships that hold, transiently, among the letter tokens and higher-level structures instantiated in specific "worlds". All Platonic entities (types) reside in our concept network, the Slipnet, whereas all ephemeral entities (tokens) reside in the "cytoplasm" (both of these are described below).

For example, such concepts as "first", "last", "successor", and "predecessor" apply timelessly to letter types. Thus A is the first letter of the Platonic alphabet, its successor is B. Z is the last letter, and has no successor (thus the Platonic alphabet is not circular). On the other hand, transient structures such as RQQQX involve a different kind of ordering, straightforwardly called "left-to-right". Thus, the R here is a letter token, and is called *leftmost* (not first); its *right-neighbor* is a token of type Q. There is only one Q type, but there can be any number of Q tokens, as in RQQQX. Although letter tokens, not being Platonic entities, have no successors, they can have *successor-links* to other letter tokens. Such links could attach any or all of the three Q's to the single R, but they are not automatically present. They are for the program to insert (or remove) as it sees fit. On the other hand, neighbor-links, being the threads that knit an ephemeral structure together, are automatically present, and they remain for the lifetime of any ephemeral structure.

We believe that a prerequisite to making good analogies in any domain is a repertoire of higher-level (semantic) constructs based on more primitive (syntactic) perceptual connections (links) between atomic entities. Usually, syntactic links join just two atomic entities, and the simplest semantic units are "chunks" formed from an arbitrary number of syntactically-linked atoms. Such units themselves can then be syntactically linked, and thus hierarchical perception can take off. We have formulated what we believe to be the most fundamental semantic concepts in the alphabetic micro-world (and psychological research by **Restle** has confirmed our intuitions). They are as follows:

Definition: A *C-group* (copy-group) consists of any nonnegative number of copies of any structure.

Examples: AAA, KK, PQPQPQPQ, M.

Note: C-groups have two parameters, or *salient roles*: the number of copies, and the structure copied.

Definition: An *S-group* (successor-group) consists of any nonnegative number of letter tokens in left-to-right sequence whose types occur in that order in the Platonic alphabet.

Examples: ABC, XYZ, PQRS, IJKLMNOP, FG, T.

Note: S-groups have two salient roles: the earliest and latest letters (coinciding with leftmost and rightmost).

Definition: A *P-group* (predecessor-group) is a backwards-running S-group.

Examples: CBA, ZYX, SRQP, PONMLKJI, GF, T.

Note: P-groups have two salient roles: the earliest and latest letters (coinciding with rightmost and leftmost).

Again, the type-token distinction can be made here. The *concept* of C-groups is a Platonic concept, and there is but one such concept, represented by a node in the Slipnet. Of course, there may be any number of instances, or tokens, of this concept, represented by ephemeral nodes in the cytoplasm.

These are, of course, domain-specific relations and concepts. Yet, since they are very simple, we feel that they have a generality that transcends our alphabetic domain. For instance, the abstract concept behind C-groups is obviously that of constancy, repetition, uniformity in texture, and so on -- clearly a critically important notion in any sense such as vision or hearing. The abstraction lurking behind the concepts of S-group and P-group is simply that of change, especially uniform change, such as is manifested visually in linear motion, or auditorily in a scale. Constancy and uniform change are so fundamental that we feel C-groups and S-groups are nearly domain-independent notions. (For a simple extension of the notions, see **Hofstadter:OSW**).

There are, however, other notions that are even more domain-independent. An example is *salience* (i.e., being a distinguished element). For instance, A and Z are distinguished elements of the Platonic alphabet. What this means is that it is generally desirable to cast descriptions of structures in terms of them. Thus, if a B occurs in a structure, it may be better to view that letter as an instance of "successor-of-A", rather than as a mere B. This means that a small amount of salience is conferred upon Platonic B merely in virtue of its being the successor of Platonic A (reminiscent of the reflected glory enjoyed by the best friend of the most popular kid in school). However, there is a gradual decay of salience, in that the more distant the connection with the inherently salient object, the weaker the transferred salience. By the time you reach Platonic D, the effect is very weak if not totally gone. (The reason we cannot say exactly where the effect vanishes totally is because of the probabilistic nature of the spreading activation in our Slipnet, to be discussed at length below.)

Another notion applying to structures of most interesting domains, and which often plays a significant role in analogy-making, is *symmetry*. Symmetry exists at many levels of abstraction. Typical examples of symmetry at various levels are: ABA, ABCTTTABC, ABCXYZ, ABCZYX. The first two are *syntactically* symmetric (i.e., knowledge of the alphabet is not required for the symmetry to be apparent). ABA is symmetric at the letter level; ABCTTTABC at the group level. The next two involve "semantic" symmetry, i.e., symmetry based on mirror-imagery inside the semantic space of the alphabet itself (A being the mirror image of Z, and so on). ABCXYZ is semantically symmetric at the letter level (that is, its first and last letters are semantic mirror images, and so on), while the semantic symmetry of ABCZYX is at the group level (the S-group ABC is the semantic mirror image of the P-group ZYX). For a structure to exhibit symmetry is really a type of analogy itself, in that the structure maps onto itself, at some level of abstraction.

The permanent knowledge base of Copycat consists of knowledge about (1) the Platonic alphabet; (2) how ephemeral structures are built; (3) how ephemeral structures can be annotated by links; (4) abstract concepts such as C-groups, S-groups, and P-groups; (5) connections among abstract concepts (e.g., "S-group" and "P-group" are "symmetric-opposites"); (6) connections among such connections (e.g., the link between "successor" and "predecessor" is of the same type as that between "right-neighbor" and "left-neighbor"), and so on. All of this knowledge is encoded in our

Slipnet. described in more detail in the "Methodology" section. It should be mentioned that distinctions between letter classes (e.g., vowels and consonants) are not known to the system: however, it could be an interesting extension to add some such distinction.

ANALOGIES IN THE COPYCAT WORLD

The problems to be solved are all of this type: If we have two structures X1 (the *prototype*) and X2 (the *target*), and we modify X1 somehow, producing a new structure X1* (the *result*), what would be "the same" modification of X2? That is, what is X2* (the *goal*)? We call the remodeling action that converts X1 into X1* the *change* or *visible action*. Schematically, it looks like this:

```
visible action:     X1 (prototype)  ==>  X1* (result)
                    ------------------------------
                    X2  (target)    ==>  X2*  (goal)
```

The idea of doing "the same thing" to an entirely different object, perhaps one with a quite different structure, is what gives this project its name of "Copycat". A description of the visible action, abstracted out of it so that it can be "exported" to other targets, and used to justify an answer, is called a (*perceived*) *rule*.

Suppose, for example, that X1 is the simple sequence of letters ABC, and that X2 is PQR. If X1 changes to ABD, what should X2 change to? Note that we did not say, "If the C in X1 changes to a D", for such phrasing, by telling you how to conceive of the visible action, would destroy the validity of the exercise. Thus, rather than describe the change, we simply exhibit it and let it speak for itself:

```
ABC  ==>  ABD
-------------
PQR  ==>  ???
```

We have tried this out on many people. Most people assert, quite quickly and confidently, that PQS seems a good choice here (probably the best), having inferred (mostly unconsciously) that the rule is "Replace the rightmost letter by its successor". Only a few people suggest PQD, whose rule would presumably be "Replace the rightmost letter by D". Some intelligent people do, however. Fewer still suggest PQR, whose rule would likely be "Replace all C's by D's."

Why do certain rules dominate other rules? Can one characterize the dominant rules easily? It might appear that the most abstract (in some sense) rule prevails. But is that always the case? By no means, as we shall see below. In fact, we contend that no single fixed rule will suffice -- in other words, there is no definitive formulation, no matter how elaborate or subtle it might be -- a surprising claim that would take us off course to defend theoretically (but see **Hofstadter:MMM**), but which is at least made plausible in a series of examples below.

Suppose we consider PQRS as our target, instead of PQR. There are certainly many reasonable (or semi-reasonable) answers. Let us look at some of them, roughly in descending order of quality. (We make no claim of completeness or objectivity!)

```
ABC  ==>  ABD     (change, or visible action)
-------------
PQRS ==>  ???
```

PQRT (Change the rightmost letter to its successor.)

(This is certainly the most favored answer, hands down.)

PQRD (Change the rightmost letter to D.)

PQDS (Change the third letter to D.)

PQSS (Change the third letter to its successor.)

PQRS (Change all C's to D's.)

(It is not easy to choose which is the best of the preceding four answers.)

PQRS (Change any letters occurring immediately to the right of B's to D's.)

PQRS (Change any letters occurring immediately to the right of B's to their successors.)

(Note that although the last three answers are all PQRS, their justifying rules are nonetheless rivals, since in the case of other targets, they would disagree with one another.)

QRST (Change all letters coming after B in the Platonic alphabet to their successors.)

PQD (Replace everything after the first two letters by D.)

(This seems very heavy-handed.)

PQBD (Change the rightmost two letters to BD.)

(Even more heavy-handed.)

ABD (Change the target structure lock stock and barrel to ABD.)

(Notice that while this seems arbitrary, one could well ask -- playing devil's advocate -- why replacing a large structure should be seen as more arbitrary than replacing a smaller one, such as a letter.)

All of the preceding analogies involved just one fixed target, PQRS. Although PQRS is not isomorphic to ABC, its structure is easy enough to map onto that of ABC. But other targets can present much subtler mapping problems. With some, it will be clearly wrong (maybe even impossible) to transfer the suggested rule literally -- even if it is very abstract. What seemed a clear role in the prototype may simply not exist in the target. It is at this point that the attempt to map "alien worlds" onto each other becomes crucial. One searches for new, possibly more abstract, ways of conceiving the two structures. so that unseen roles emerge from the gloom. Then one tries to map these roles onto each other, to establish a solid mapping of target onto prototype. In a sense, the act of "worlds-mapping" creates stresses on the rule, causing it to "buckle" or change itself in some way -- to "slip", as we say. Perhaps it even forces a re-examination of the grounds for the original rule.

The following series of targets is critical to our report, in that it demonstrates the enormous range of pressures that unanticipated targets can provide. In some ways it parallels the escalating series of "monsters" given by Lakatos in his work *Proofs and Refutations* (**Lakatos**), revealing how the formal notions proposed by mathematicians always fall short of the full richness of internal imagery and intuitions. Here, we show how any prior certainty about what the rule "must be" is easily violated or

cast in doubt by some new target that blurs the categories in unexpected ways. Each sample target is accompanied by some commentary on the pressures it brings to bear.

CDE ==> ???

There is some pressure to change E to F, producing CDF. But this is opposed by pressure urging that the C be changed to a D (and the rest be left alone), thus producing DDE. The idea "change C to D" is taken more seriously now than with targets PQR and PQRS, because here we have a concrete C to deal with (especially one at the boundary of a structure).

An attachment to a particular letter type, such as C, rather than to some role played by the token in the prototype, such as "rightmost", will henceforth be called *extensional*, while role-based views will be termed *intensional*. And roles whose descriptions involve concepts with higher semanticity will be considered to be "more intensional" or "more semantic" than roles whose descriptions involve concepts with lower semanticity. (See **Hofstadter:SHK** for a discussion of levels of intensionality.)

ABCD ==> ???

Similar extensional-intensional fights are set up here, except that they are exacerbated somewhat by the fact that the entire prototype ABC is contained within the target structure, thus rendering it very tempting to simply copy what was done before, thus producing the literal-minded ABDD instead of the more intensional choice ABCE.

XCT ==> ???

The serious pressures to change C into D are now somewhat modified, although not diminished. Suddenly it is brought home to us how important a characteristic of ABC was its "S-groupness", heretofore unmentioned (although certainly noticed unconsciously). The solutions XDT and XCU are both tempting, but to different parts of us.

PQC ==> ???

A subtly different blend of pressure-flavors. Change C to D, or change Q to R? Remember that PQ is an S-group, albeit a short one.

AAABBBCCC ==> ???

The obviously most desirable answer is AAABBBDDD, despite the fact that we are changing a whole group of letters, without any precedent at all.

AAABBBKCC ==> ???

Do we change only the last C to a D? Both C's to D's? KCC to DDD, or to LDD? Why? This is a particularly nasty example.

PXQXRX ==> ???

Of course, PXQXRY is possible, but it seems too simple-minded -- too syntactic. A more semantic choice is PXQXSX, but it encounters some competition from PXQXSY, which has a strange appeal. Then there is PYQYRY, whose justification is "change the latest letter (in terms of the Platonic ordering) to its successor". We will not even mention a myriad of other answers that theoretically could be proposed.

AABABC ==> ???

All depends critically on how you perceive this odd structure. If you subdivide it as AAB-ABC, then it doesn't make much sense. You might be inclined simply to change the

ABC into ABD, making AABABD. But what if you perceive *three* groups: A-AB-ABC? This is a sequence of three S-groups: A through A, A through B, A through C. Notice that if you factor out the repeated phrase, you have just A,B,C. The immediate flash is to change the final C of this derived structure to a D. This would mean "A through A, A through B, A through D", or A-AB-ABCD -- a much more abstract and satisfying answer than AABABD.

AZA ==> ???

The bounces between A and Z set up an image of a "world" in which "the next" means Z if we're at A, and vice versa. Since the visible action seems to involve skipping one step, we want to jump from Z not *to* A but *over* A, to Z, thus making: AZZ. This is quite a conceptual leap, yet it is very appealing, if not compelling.

RQP ==> ???

The fight here is between syntactic (left-to-right) and semantic (alphabetic) order. Which dominates? If the former, we will want to alter the P; if the latter, the R. But whichever choice we make, we will face the question: How? If we decided to alter the R (a semantic victory), it would seem more pleasing to replace it by S; yet if we decided to alter the P (a syntactic victory), it would seem more pleasing to replace it by O (which involves the semantic "reflection" of "successor" into "predecessor"). These are tricky and delicate and subjective matters. But what is objective is that the "backwardsness" here creates pressure to make the rule modify itself accordingly (i.e., to slip).

XYZ ==> ???

Sudden unexpected blockage of the simplistic recipe "Change the rightmost letter to its successor" occurs, because Z has no successor. What alternative routes are open to us? Since this example will be considered later in the "Methodology" section in some detail, we postpone discussion of it until then.

Some of the answers above are appealing while others are appalling. Why is this so, in such an abstract, seemingly content-free and connotation-free domain? It is because we cannot turn off our judgment-making machinery, even when we know we are in an artificial situation where our judgments will not affect our survival or our well-being in any way. The mechanisms of perception, grouping, assessing relatedness and relevancy, deciding what level of abstraction something belongs to -- all these mechanisms remain dominant mental forces even when we realize we are operating in an artificial domain. In such a domain, in fact, these mechanisms emerge particularly clearly, unobscured by domain prejudices. At least this is a critical article of faith underlying our choice of this domain.

METHODOLOGY

Our methodology is based upon the progress made so far in developing the architecture of the prototype system called "Jumbo", and upon a set of ideas more specifically tailored to the problem of analogies and judgments. We shall first describe the Jumbo system, developed mostly in 1982, and then we shall describe how we envision building Copycat, using some ideas in Jumbo and newer ideas about the slippability network and how activation spreads in it. (See also **Hofstadter:JUM**.)

I. Jumbo

The Jumbo project was undertaken expressly as a prelude to the more ambitious Seek-Whence, Copycat, and Letter Spirit projects (see **Hofstadter:MT1**, **Hofstadter:OSW**, and **Hofstadter:MMM**), and its purpose was twofold:

(1) to study a specific type of parallelism inspired by the Hearsay II speech-recognition project as well as by the parallelism of enzyme-mediated activity in the living cell; and

(2) to study the nature of data structures that can be fluidly put together, taken apart, and restructured internally, while maintaining "semantic stability", which means that a structure, once created as a member of a particular class, will retain the characteristics that qualify it for membership in that class, provided the restructuring it undergoes is not too severe.

The domain of Jumbo is that of the newspaper puzzle called "Jumbles". In such puzzles, the player is asked to rearrange a given set of letters to form an English word. Usually about six letters are involved, but the exact number does not matter. The Jumbo system simulates one part of this playful activity: that of synthesizing plausible word candidates out of the given "raw materials" (letters). The program is not endowed with a dictionary of English and thus cannot know whether it has succeeded. The reason for providing no dictionary is not to frustrate the program but to have it judge its progress on purely internal criteria of coherency at several levels of structure at once. If it succeeds in creating a good candidate, it will display it (and of course all intermediate stages in the work can be displayed as well). Knowledge of English consonant clusters and vowel groups is built in. The main task for the program is to bring this knowledge to bear on the situation at hand.

While the choice of Jumbo's domain might on the surface seem frivolous, actually it was motivated by a lifelong fascination with the unconscious processes of rapid assembly, disassembly, and regrouping of letters that seem to take place at lightning speed in the unconscious mind. These phenomena are not restricted to the manipulation of letters, of course. They take place in the manipulation of ideas of all sorts, particularly in creative activities such as writing, composing music, making mathematical discoveries, tossing ideas about in search of a new way to understand them -- and, of course, in making successful analogies. This playful rearrangement activity might be called "fussing around with ideas", a humorous term for one of the most central processes in the generation of new ideas. The goal of understanding the nature of swift, unconscious "fussing around" (structural manipulation) was the *raison d'être* of Jumbo, since it was felt that when such operations were taken collectively and controlled efficiently, they could provide the substrate out of which would emerge the desired fluidity.

CELLS, ENZYMES, AND PARALLELISM

Operations in Jumbo can be placed in three classes: entropy-decreasing, entropy-preserving, and entropy-increasing. The first corresponds to activity in which structures are being assembled, the second to activity in which in which structures are being regrouped or restructured, the third to activity in which structures are disassembled. For each kind of activity there are specific pieces of Lisp code, usually referred to as *codelets*, but occasionally referred to as *enzymes*, because of the all-pervading influence of the metaphor of the analogous processes inside living cells.

It is worth making a one-paragraph digression on the biological metaphor, in fact, since it plays such a fundamental role. In a cell, all activity is carried out by enzymes. Enzymes of various sorts are distributed at random throughout the *cytoplasm* (the cell's interior), and because of random motion taking place inside the cytoplasm, they encounter all sorts of molecules in a very short time. Each enzyme has one or more (usually two) active sites -- physical slots that fit a specific type of substrate, or molecule. When an enzyme encounters a molecule that fits one of its active sites, it latches onto that molecule and fills that site. When all of its active sites are filled, the enzyme then performs its function, which may be constructive (combining two substrates into a larger molecule), reconstructive (changing the structure of a substrate), or destructive (reducing substrates into their components on a lower level of molecular structure). Usually, one enzyme's action is but a small link in a long *chain* of enzymatic actions whose collective result is the buildup of some complex product, such as an amino acid, a nucleic acid, a protein, a chain of DNA or RNA, and so on. It is to be emphasized that the cell relies on the random peregrinations of molecules inside it for these activities to be carried out; there is no Director General who observes all and shunts all pieces to their proper places at the proper times. This is simply because a cell is at too low a biological level to have such a centralized intelligent agent. A cell's "intelligence", such as it is (and it is astonishing!), must emerge from the interplay of thousands of small, independent processes whose outcomes have effects on the further activities to take place. A cell's "intelligence" is of necessity highly distributed, in short, with wave after wave of enzymatic activity creating order out of chaos. In particular, the products made by one set of enzymes become the substrates to another set of enzymes. One remarkable feature of the cell is that enzymes themselves are produced, altered, and destroyed by other enzymes, so that the enzyme population of a cell is incredibly sensitive to the "needs" of the cell: it is constantly adjusting itself according to the types of substrates present and absent. Elaborate feedback loops regulate the enzyme population. This rich metaphor has repeatedly furnished ideas for the design of the Jumbo and Copycat projects.

The locus of all structure-changing operations in Jumbo is called, naturally, the *cytoplasm*. For purposes of orientation, it could be compared to the Hearsay Blackboard (see Reddy and **Erman et al**), although there are notable differences. The initial contents of the cytoplasm is the set of letters to be "fussed around with". Gradually, the actions of codelets combine letters with one another and transform the cytoplasm into a smaller number of pieces, and eventually the cytoplasm will wind up with just one large piece, a word candidate. The generic name for pieces of all sizes (usually larger than the single-letter size, but not always) is *glom* (coming from the colloquial verb "to glom together").

There are codelets for combining consonants into clusters, vowels into vowel groups, consonants and vowels into syllables or syllable fragments, syllable fragments into full syllables, and finally, syllables into polysyllabic structures, or word-like objects. These are the constructive, or entropy-decreasing, operations, of course. There are likewise codelets for operating on structures from the inside and transforming them internally -- the entropy-preserving operations. Such operations typically perform boundary shifts ("pan-gloss" becomes "pang-loss", for instance), or more complex

shifts like spoonerisms ("pang-loss" becomes "lang-poss"). Then there are destructive, or entropy-increasing, codelets, which break bonds established by earlier constructive codelets.

At any moment, many codelets are potentially able to act on the contents of the cytoplasm. In a fully parallel system, several could actually act at once. In the model, one must run at a time. However, this does not reduce the effective parallelism. The cell's processes, it must be emphasized, are not the actions of single enzymes, but rather the long *chains* of activity of many enzymes in a row (the Krebs cycle and the photosynthesis cycle are typical examples involving dozens of reactions apiece). Thus if one allows one enzyme from Chain 1 to run, then one from Chain 2, one from Chain 3, and so on, in a time-sharing manner, all three chains will proceed toward completion in parallel. It is in this sense that Jumbo is a deeply parallel system. It is not at the codelet level, but at the codelet-chain level, that Jumbo's parallelism becomes apparent.

The "operating system" that timeshares codelet chains is not aware of the chains as such. It sees only codelets, and makes its choice at that level alone. In fact, codelet chains are not explicit elements of Jumbo in any sense; a codelet chain is in the eye of the perceiver. Long sequences of codelet actions are manufactured dynamically, with one codelet producing any number of successor-codelets, thereby attempting to extend its own computational line.

RANDOMNESS AND THE PARALLEL TERRACED SCAN

At this point it is most important to emphasize one critical and far-reaching decision made concerning Jumbo's scheduling algorithm: it is random -- not chaotic, in the sense that all codelets are equally likely to run at any time, but random in a carefully controlled way. There is a data structure called the *coderack* upon which are hanging ready-to-run codelets (these can be thought of as quoted function calls). Each such codelet, when hung up on the coderack, is assigned an *urgency*. This is an integer that determines the codelet's probability of being run next. Specifically, a codelet's probability of being chosen next is the ratio of its urgency to the sum of urgencies of all codelets on the coderack.

A codelet's urgency is qualitatively quite different from the priority of a task in a queue in a deterministic agenda-based scheduling system, such as Lenat's AM (see **Lenat:AM**). In particular, high-urgency codelets are not necessarily very likely to be run next. For example, suppose that the coderack contains 90 codelets of urgency 1 each, and one codelet of urgency 10. By definition of urgency, the chance that the unique high-urgency codelet will run next is merely one in 10, whereas the chance that some low-urgency codelet will run next is 90 percent. No *particular* low-urgency codelet is likely to run next, but since many of them are flooding the coderack, the high-urgency one is swamped, and must patiently await its call. Clearly, a probabilistic scheduler like this is not based on a policy of always letting the "best" codelet run first. Rather, such a system effectively explores many routes in parallel, multiplexing them at the codelet level, and advancing each chain with a speed proportional to the average urgency of its component codelets. By contrast, a deterministic agenda-based system explores just one avenue at a time, always trying first what it considers best. This is a very significant philosophical difference.

So far it appears as if codelets always modify pieces of the cytoplasm, like enzymes. This is not true, for there is another type of codelet, opposed to such "action" codelets, called a *musing* codelet, whose purpose is simply exploratory. The reason for this is to increase the subtlety and sensitivity of Jumbo to many different potential pathways of activity. Musing codelets allow several different and rival pathways to be "sniffed", or checked out, to varying degrees of depth, without any action (i.e., modification of the cytoplasm) necessarily taking place. Musing codelets serve the function of preliminary study committees, or scouts. This means that a type of progressive deepening of

exploration can be integrated into the system.

We call this a *parallel terraced scan*; it is a way of ensuring that the system will tend to explore many pathways in parallel, with speeds roughly proportional to their promise. This is a central ingredient of the Jumbo architecture, and owes much to Hearsay II (see APPENDIX 3). The parallel terraced scan works by having the system constantly monitor the progress along the different avenues and set the urgencies of new codelets accordingly. In particular, promising-looking opportunities can be assigned higher urgencies, not only for action, but also for musing. Less promising avenues can be explored, but their priority of exploration will be lowered, so that on the average, usually only the best routes will be scouted carefully, but of course there is no ironclad guarantee of that, and so occasionally, a quirky but still reasonable pathway may be the one followed. The low-level randomness of the Jumbo system is a critical element of the parallel terraced scan; it allows many explorations to proceed at once even if some appear more promising than others. Our strategy is closely related to the optimal strategy for the "two-armed bandit" puzzle (see **Holland:ARM**).

Musing codelets are like the fingers of a lake whose water level is rising: tentative forays go out in many directions simultaneously, but only the first to "discover" a place where the water can plummet downwards will actually change the course of flow. (The downhill flow of water corresponds to cytoplasmic actions taking place.) But even after this happens, this does not commit the lake exclusively to that one "chosen" pathway. There are now both inflow (from above) and outflow (out the tip of the "lucky" finger). If the water level is still rising (i.e., the inflow still exceeds the outflow), that will cause the other fingers to stretch still further out, even while new fingers or "subfingers" are sprouting -- and eventually water will gush over one of them as well. This process will continue until the total outflow exceeds the inflow, so that the water level is no longer rising, and the source pushing forward this many-pronged parallel exploration is removed. In Jumbo, the fingers of exploration are the chains consisting of musing codelets. Such a chain can involve several probes, at increasing depth, of a given potentiality. The rates of progress of the various probes are all dynamically controlled by the results they uncover along the way. The way this happens, to spell it out, is that a given test will not only place its successor test in the coderack, but will also determine the urgency of that new codelet, on the basis of its own findings. Thus there is constantly being carried out a parallel revision of the promise offered by the rival pathways.

RESTRUCTURING AND DESTRUCTIVE OPERATIONS

If assembly were the only possibility open to Jumbo -- i.e., if only entropy-decreasing codelets existed -- then all of Jumbo's seething parallel activity would constantly build towards a single word candidate. Clusters and groups would be built up, syllable fragments, then syllables. Suppose two excellent syllables were built but when put together, the word candidate they formed was not excellent. An example would be the syllable-pairs "poo" and "eep", which do not yield a nice word-like structure either way they are combined. Neither do "oil" and "he", or "eth" and "tha". Examples aside, the point is that purely forward motion can lead you down blind alleys from which there are no escapes. In particular, if you can never undo actions taken, you may find that you are painting yourself into corners. That is the problem with such a purely constructive, or entropy-decreasing, approach. It always improves things locally, taking what has been built up so far and building upon it further, but never dismantling anything. For a while one seems to be getting closer to a word candidate, but then, when the critical moment of assembly at the word level arrives, one finds that no combination works. Or, one might find that some combinations work, but only very weakly. Still, there is no backing out because no other types of action codelets exist. (Of course, one can paint oneself into a corner at any level, not just the top level.)

Such a solution technique is known as "iterated improvement", and its hallmark is one-way motion toward a solution, without any possibility of undoing any actions taken. In order to allow the search space to widen out again, we have introduced, in Jumbo, other types of action codelets: those that keep entropy constant, and those that increase it.

Typical entropy-preserving codelets (and the term "entropy" is used here purely figuratively) are ones that involve transformations of existing structures. Such operations as flips, swaps, merges, splits, and shifts are allowed. A *flip* is where a structure (e.g., a syllable) is reversed. Thus, "top" becomes "pot". A *swap* is where two structure exchange pieces. Thus, "wash-cloth" becomes "clash-woth" or "wath-closh" or "wosh-clath". A *merge* is where two adjacent pieces are reperceived as one piece at the same level. Thus, the two syllables in the word candidate "blo-at" could be merged into the single syllable "bloat", which becomes a new word candidate, automatically. A *split* is the inverse operation, in which one unit at some level splits into two units at the same level. A *shift* is where an internal boundary between lower-level units is dislocated, as in the shift from "winth-rop" to "win-throp", or "sun-glasses" to "sung-lasses". It is noteworthy that such operations resemble those that take place in genetic recombination. In fact, John Holland, in his work on self-organizing intelligent systems, has used some of these same operations acting on genome-like data structures to show the power of recombination, together with evolutionary selection, to evolve highly efficient systems very swiftly (see **Holland:GEN**).

Despite the power of entropy-preserving transformations, there may be occasional reason to despair of the progress made so far, to feel that one is in entirely the wrong region of the space, and that more radical revision is needed. For this purpose, a third repertoire of action codelets is provided, the "breakers" (as opposed to the "makers"). The purpose of a breaker is, of course, to destroy a structure and thus allow new pieces to glom together spontaneously. If the coderack is loaded up with many breakers, then what has been built up will degrade rather quickly into its ultimate constituents at the letter level. On the other hand, one can be more delicate about it, and insert specific breakers targeted on specific gloms, so that much of the built-up structure is left intact, while some especially bad or troublesome piece is destroyed. Needless to say, musing codelets can "scout" ahead along breaking pathways just as easily as along making or regrouping pathways; there is no more impulsiveness necessarily involved in this type of activity than in any other type.

HAPPINESS, TEMPERATURE, AND SELF-WATCHING

The carefully controlled interplay of makers, regroupers, and breakers can provide a powerful and swift method of converging on a high-quality word candidate. But in order to control this interplay sensitively, one needs at all times to monitor the progress being made, and to control what types of enzyme are allowed to enter the coderack. To aid in evaluation of progress, there is a numerical measure of *happiness* that applies to every glom on every level. There are two components of happiness: internal and external. The *internal* component for, say, the syllable level, involves knowledge of what a "happy" syllable looks like. Ideally, though not necessarily, a syllable begins with and ends with a consonant or consonant cluster, and contains a vowel group between them. The vowel group is, of course, necessary, while the other two are dispensable. Ideally, each of these three pieces is in itself a "happy" glom (where "happiness" at the consonant-cluster or vowel-group level also breaks up into internal and external components -- and so on). The *external* component of happiness of any glom depends on two things: (1) whether the glom is already incorporated into a larger glom, and (2) if not, the prospects for such incorporation. (No glom will be happy if it is internally flawless but unusable.) A numerical measure for the happiness of any glom can be computed from these various components, and by watching the overall happiness of the gloms in the cytoplasm, the system can get a good feel for how well it is progressing towards its goals.

As long as there is significant improvement in the happiness level, or as long as the happiness level is adequate at the word level, there is no need to employ any enzymes other than constructive ones. But if one gets stuck in a situation where two syllables strongly resist combination into a word, or where the best word candidate is unusually weak, then radical measures may need to be taken. At this point, if one can clearly point at "defective" gloms responsible for the trouble, one can insert specifically targeted breaker gloms to attack and destroy those gloms.

However, as often as not, the blame will not be localizable to one or two gloms. In that case, there is recourse to altering the system *temperature*, a variable regulating the amount of disorder to be deliberately injected into the system to render its search for a good state more efficient. The value of the temperature is derived from the happinesses of the top-level gloms in the cytoplasm. It is highest when the top-level glom is least happy, which is to say, most in need of an escape route from what is apparently an unfavorable region of the search space. Such an escape route is provided by the introduction into the coderack of musing codelets scouting ahead for breaking enzymes. The temperature is lowest -- approaching freezing, or zero, when the top-level glom is deemed satisfactory by all criteria known to the system, so that there is no need to seek any radical alternative solutions. By carefully monitoring its own happinesses and regulating its temperature, Jumbo can thus "unpaint" itself out of a corner when needed.

One type of self-watching that was thought about for Jumbo but not implemented involved the system watching its own coderack and cytoplasm, looking for loops in behavior in either structure, and whenever they were detected, taking remedial action. Another type of self-watching considered was for the system occasionally to take a "census" of its coderack, and if it is found to be too cluttered, to carry out a purging or "garbage collection" operation, in which low-urgency codelets would be swept out to sea. These kinds of highly "introspective" operations are, we feel, central to the workings of any intelligent system, but we did not add them to the Jumbo system, feeling that we had pushed Jumbo far enough and that a more cognitive task (such as analogies, for instance) was needed for us to push the architecture further. (See APPENDIX 4 for comments on various approaches to highly "introspective" systems, such as those of Smith and Lenat.)

II. Beyond Jumbo

We have now been through the architecture of a system designed to create "well-chunked wholes" from initially scattered pieces, and to allow such wholes to regroup themselves fluidly into many different configurations. The system tries, under many constraints, to achieve a good overall balance. In Jumbo, the constraints had to do with making "happy" structures at several levels simultaneously. Each assembled structure was a member of some category (cluster, syllable, etc.), and in general it was always clear what category was involved. In Copycat, the goals are somewhat similar: we are given a "raw" cytoplasm containing ephemeral structures but no syntactic links or semantic chunks and we want to create such links and higher-level chunks and then to be able to regroup them fluidly so as to make maximally "happy" descriptions.

But there are two obvious differences between Jumbo and Copycat. First, the basic elements in the Copycat cytoplasm (the letter tokens making up a particular analogy problem) are presented in a specific left-to-right order, and so the problem is not to figure out which ones might make good neighbors, but where to draw boundaries, what kinds of categories to see structures in terms of, what kinds of cross-compartmental connections to draw, and so on. The second difference is that Copycat has a much richer category system, and so there are going to be strident fights for describing structures and roles within them. Even in so simple a structure as AB, the B can be perceived as (1) simply one token of the letter type B, (2) the successor of the leftmost letter of the structure, (3) the

right-neighbor of an A (salient because A is the Platonically first letter type), (4) the rightmost letter, (5) the second letter of the structure, (6) a C-group containing one element, and on and on.

With so many possibilities, it might seem very hard ever to settle on a single view of any structure. Indeed, that seems to be at the crux of the objection to the analogy problem as offering any fundamental insight into intelligence, since it all seems so subjective and arbitrary, at least on a superficial glance. However, it is a fact that perception (i.e., good description) leads to consequences in life, and in particular to differential rates of survival, and therefore it must be concluded that certain ways of choosing descriptions must be more efficient, more attuned somehow to the way the world works -- and it is those that we feel humans have par excellence. We feel that humans are at the pinnacle of good recognition, perception, categorization, abstraction -- even if it is subjective.

In order to simulate this capacity in Copycat, we must instill sophisticated guidelines for choosing what to single out as important in a structure, and what to neglect. There must, in other words, be a *topography* to the space of concepts, something that makes it non-flat, so that general preferences are established, and so that guidelines exist for altering descriptions rapidly to satisfy more preferences at once. This is a major part of the task of Copycat, and we would like to sketch how Copycat can carry this out. There is no better way of doing so than going through one example that illustrates many of the central features of Copycat, as we now envision its eventual implementation.

As our canonical example of the mode of functioning of Copycat, we shall therefore focus on the handling of the following analogy:

```
ABC ==> ABD
-----------
XYZ ==> ???
```

We shall show a plausible (but by no means unique) route to the plausible (but by no means unique) answer WYZ. To sketch it out in detail would be excruciating and would afford such a good view of the trees that we fear the forest would become invisible. Therefore we shall attempt to make the forest as clear as possible, at the sacrifice of detail.

In order to show where the crucial junctions occur, we shall compare the places where the treatment of this particular example would diverge from that of two other closely related analogies:

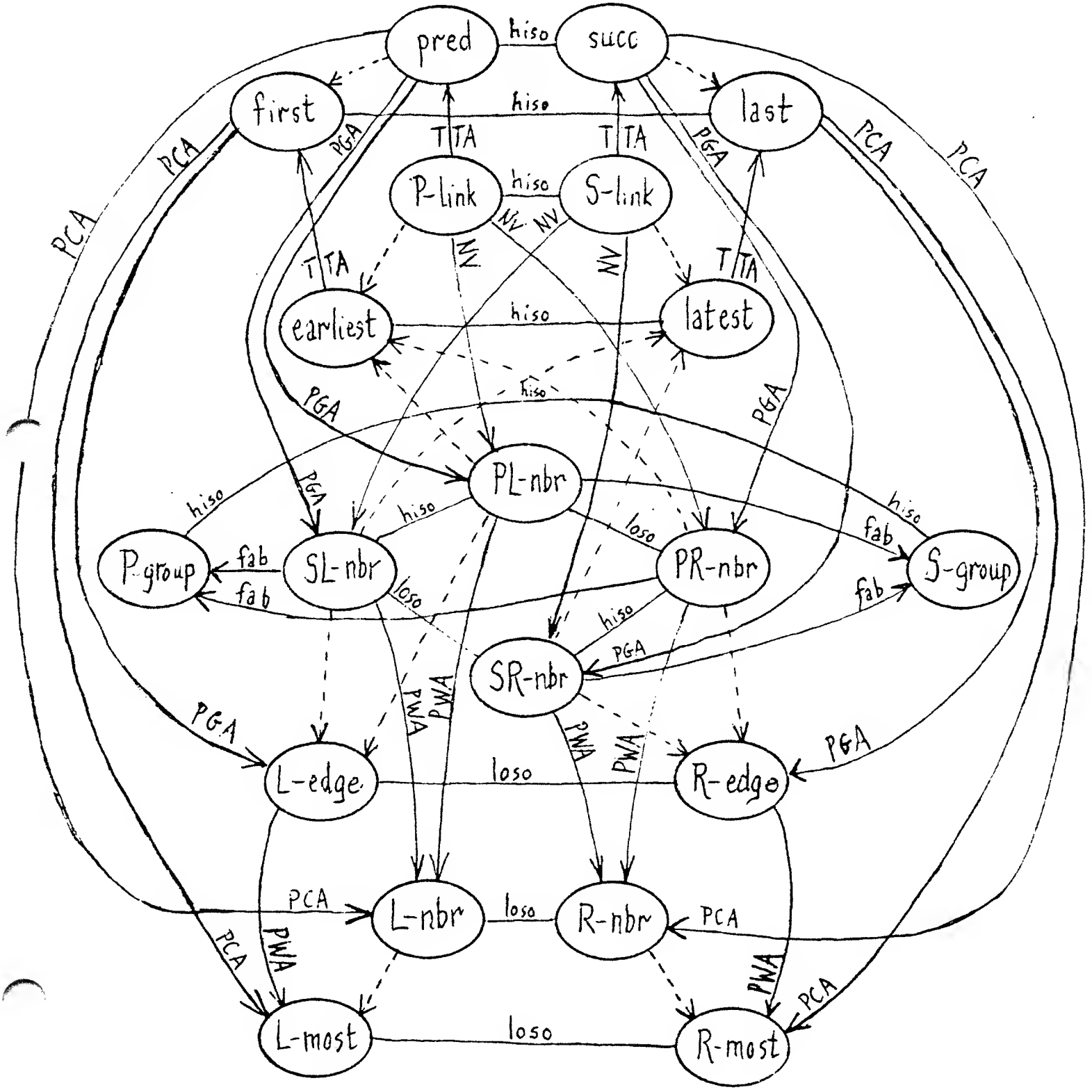
```
ABC ==> ABD
-----------
PQR ==> ???
```

```
ABC ==> C
---------
XYZ ==> ?
```

Each of these is simple and invites an obvious answer. The first one cries out for the answer PQS, as we observed earlier. The second seems to cry out just as loudly for the answer Z. Together they will help put our example in perspective.

To give the punch line away (but not too much), here is what these contrasts show. The first one says, in effect, "The change ABC = => ABD should be interpreted as taking the successor of C." The second reconfirms one's intuitions, in case there ever were any doubts, about what the counterpart of C in XYZ really is. In effect it says: "In comparing ABC with XYZ, Z is surely the counterpart of C." Put these two together, and you are led straight to taking the successor of Z -- an impossibility in our world! So something's got to give. That is why this example is particularly interesting. (In APPENDIX 5 we present a systematic set of variations on our basic analogy and discuss what they reveal about this analogy and the role that intensionality plays in it.)

...at part of the Slipnet dealing with Platonic and cytoplasmic adjacency



Nodes

pred:	predecessor
succ:	successor
P-link:	predecessor-link
S-link:	successor-link
PL-nbr:	predecessor and L-nbr
SL-nbr:	successor and L-nbr
PR-nbr:	predecessor and R-nbr
SR-nbr:	successor and R-nbr
P-group:	predecessor-group
S-group:	successor-group
L-edge:	left edge (of group)
R-edge:	right edge (of group)
L-nbr:	left neighbor
R-nbr:	right neighbor
L-most:	leftmost
R-most:	rightmost

Link-types

loso:	low-level (syntactic) symmetric-opposite
hiso:	high-level (semantic) symmetric-opposite
fab:	fabric (of structure or substructure)
----→:	iteration results in
NV:	neighborized version
PGA:	Plato-group analogue
PCA:	Plato-cyto analogue
TTA:	type-token analogue
PWA:	part-whole analogue

THE SLIPNET -- SOURCE OF FLUIDITY IN COPYCAT

Before we actually describe how Copycat might tackle any analogy problem, we must describe its category system -- the Slipnet -- in some detail. (See the figure.) The Slipnet might be called a "semantic network", but if so, it is not in the traditional AI sense of the term. To be sure, the semantics of all domain concepts resides (explicitly or implicitly) in the Slipnet, but the Slipnet is not a locus of node- and link- creation and destruction (as is the cytoplasm, or Hearsay's Blackboard). Its nodes and links are permanent, and they form a storehouse of conceptual proximities (slippability links) and semanticities (centrality values). (See **Hofstadter:GEB**, Chapter XIX.)

The only way in which activity takes place in the Slipnet is that each node has a degree of *activation*, which starts out at zero and can vary during the course of the processing. The activation level reflects the system's current "interest" in this node, in terms of propensity to use the concept in building descriptions or rules. (See **Lenat:AM** for a related concept of "interestingness" that guides computation.)

Activation spreads in the Slipnet, from a node to its nearest neighbors. However, activation does not spread uniformly, for some links -- "greased" links -- are more prone to transmitting excitation than others. The amount of "grease" on a link is not fixed, but can vary just like the activation of a node. Links come in different classes (e.g., "symmetric-opposite"), and each class is represented by a node in the Slipnet. The more activated such a link-class node is, the more greased are all the links of its class. Just as with all computational events in Copycat, the actual spreading of activation in the Slipnet is probabilistic: codelets are manufactured for raising or lowering the activation levels of nodes, and they must await their turn to run, just like all other codelets. The urgency of an activation-spreading codelet is proportional to two factors: (1) the semanticity of the node from which activation is spreading, and (2) the amount of grease on the link along which activation is spreading. This encourages activation to spread rapidly into the more highly semantic parts of the Slipnet, and along links known to be relevant to the given situation. Due to the probabilistic nature of the spreading, activation spreads outwards from any node in a somewhat jerky way in the short term, but in the long run its spread will be more continuous. (See **Anderson**; **McClelland & Rumelhart**; **Norman & Rumelhart**; **Smolensky**; **Hinton & Sejnowski**; **Hinton & Anderson.**)

The more greased a link is, the more easily it transmits activation, and also the more "slippable" is the pathway it represents. This means that if Node 1 and Node 2 are connected by a greased link and Node 1 is in some way found suspect, then Node 2 is a likely candidate for replacing Node 1 in a description or rule. In effect, to put grease on a link is to temporarily enhance the association between the concepts it links. This facilitates easy slippage from one concept to the other, if called for.

In effect, *slippage* means deformation of one descriptive structure into another. Sometimes the deformation will yield a synonymous though superficially different structure (intensional slippage), and sometimes the deformation will yield a structure whose meaning is truly different (extensional slippage).

An example of intensional slippage is to convert either of the following descriptions of AAB into the other:

A-AB: "An A followed by an S-group running from A to B";
AA-B: "A C-group with two A's, followed by a B".

This is, in essence, a boundary shift of the sort that regrouping codelets in Jumbo carry out ("pan-gloss" to "pang-loss"). An example of extensional slippage is to convert either of the following rules into the other:

(1) "Replace the rightmost letter by its successor";
(2) "Replace the leftmost letter by its predecessor".

The combination of probabilistic control structure, slippage, and spreading activations, is the primary means by which we hope to imbue Copycat with a new kind of fluidity.

OVERVIEW OF ONE ANALOGY

Having described the Slipnet, we can now proceed to describe how it interacts with the codelets, to direct the gradual "understanding" of structures in Copycat's cytoplasm, and to solve our canonical analogy problem: "if ABC changes to ABD, what does XYZ change to?" We assume the three structures -- the prototype (X1), the given result (X1*), and the target (X2) -- have all been typed in, and that the system has done all the necessary preliminary processing of the letter tokens in each structure, which means setting up a compartmentalized cytoplasm with three compartments, one for each structure. Within each compartment, for each letter there is a node (a Lisp object), as well as neighbor-links encoding the left-to-right ordering of the letters. For instance, the node for the D in ABD has a link saying that it is the right-neighbor of the node for the B in that structure.

In very brief summary, here is what is going to happen. The treatment of our analogy problem will proceed in "phases", corresponding roughly to the "chains" of codelets described in the section on Jumbo. These phases will be presented sequentially, but they would not actually follow each other in such a neat, orderly fashion. Two or more phases might overlap, by having their codelets intermingle. Some phases might even occur in the reverse order from that presented below. The only thing that tends to ensure some measure of sequentiality is that some phases (e.g., the "closure-checking" phase) can be triggered only by the production of some specific type of structure, and so must wait for that to happen. For that reason, phases do tend to occur in a somewhat predictable order, and it is a convenient approximation to describe them sequentially. Thus:

1. *Syntactic Scanning Phase*: Codelets will come in, scan the various structures in the cytoplasm, insert links in various plausible spots (but not all), and thereby activate certain semantic concepts (here, "S-group" is the main one).

2. *Semantic Phase*: These concepts will then try to instantiate themselves where possible. Meanwhile, more scanning will try to establish long-distance links between prototype X1 and given result X1*.

3. *Rule-Generation Phase*: When these links are in place (again, plausible ones but not all possible ones), a rule will be formulated accounting for the visible action. The rule will be abstracted so that it is not totally extensional, but somewhat intensional. In other words, it is now somewhat detached from its original context, and can give sensible answers to simple targets other than the prototype.

4. *Worlds-Mapping Phase*: At this point, the system will move on, and try to map the prototype X1 against the target X2, using as guides both the long-distance links and the semantic structures discovered earlier.

5. *Rule-Slipping Phase*: The result of this "mapping of worlds" will determine whether and how the old rule needs to be adapted to the new world (i.e., slipped into a variation of itself). If needed and if possible, some slippage will be carried out.

6. *Rule Execution Phase*: The rule is now run, producing -- hopefully -- a good result. (It is at this phase that Copycat will stumble, trying to take the successor of Z. Then it must recover, by going back and taking a look at things that could have been slipped but weren't. It returns to the Rule-Slipping Phase, where a remedial slip is instigated and it transforms the rule into a cousin-rule, which now applies and gives rise to an answer candidate.)

7. *Closure Checking Phase*: The proposed answer (WYZ) is now evaluated and found successful, due to the fact that the vertical mapping between it and the given result (ABD) is recognizable as the "same" mapping as the other vertical mapping (the "worlds-mapping" between XYZ and ABC). If the two mappings were not recognizably the same, then closure would not have been effected, and a new answer would have to be sought, by returning to an earlier phase -- possibly even as far back as Phase 1.

The proper way to look at "decisions" of what to do next is not at the fine-grained level of "Which codelet runs next?", but at the coarser-grained level of "How will the current population of the coderack determine the near-term future population of the coderack?" At that higher level, there is far more determinism, thanks to statistics. Thus, although one can't explain events deterministically at the single-codelet level, one can say with a fair degree of certainty why one *wave* of enzymes triggered another wave, and so on.

The Syntactic Scanning Phase

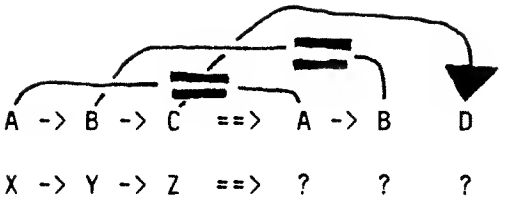
At the first stage, the most "syntactic" of all enzymes are unleashed into the cytoplasm. These are ones that look for the most primitive but "interesting" connections between immediately neighboring letters. Thus inside ABC, two successor relations are noticed and instantiated by "adjacent S-links"; similar things happen in the other two compartments, and what we end up with is shown below:

```
A -> B -> C   ==>   A -> B     D

X -> Y -> Z   ==>   ?     ?     ?
```

To be strict about it, these links are not set up without some prior checking-about. Musing codelets examine whether there is any reason to suppress any potential links. At this preliminary stage, there certainly is not.

The codelets responsible for making these purely local observations are "then" succeeded by codelets that look for *remote* links (i.e., not between adjacent letters), either within a compartment or across compartments -- either horizontally (prototype-result) or vertically (prototype-target). Putting "then" in quotes simply underlines the fact that the time-succession is not strict, but probabilistic, in that these latter codelets have been present in the cytoplasm from the start, but with lower urgencies, and for this reason, they were probably not chosen to be run very early. The most obvious remote links here are two sameness links and one successorship link linking X1 with X1*:



Naturally, the codelets that have the highest urgencies will be the first to make observations of this sort. Therefore it is important to make sure that codelets that look for simple, "natural" relationships are assigned high urgencies. But what kinds of abstract criteria are there for saying which potential inter-compartmental links are "natural"? Well, if the two objects are located in the "same position" in their respective compartments, that is good; also, everything else being equal, sameness links are better than successor-links or predecessor-links. (For two objects in different compartments to be in the "same position" as each other is, potentially, as tricky an analogy problem in itself as any that can arise in the full domain. However, often there are very crude intensional guidelines that can be used to assign urgencies to codelets in such a way that very good performance is obtained. Simple things like comparing the extremities of both compartments are good for starters.)

There are other criteria for "naturalness" of remote links, and they can be used by urgency-assigners and musing codelets to give the edge to codelets searching for "good" connections across cytoplasmic compartments. This way, "natural" connections will have a tendency to crop up swiftly. By contrast, slower (lower-urgency) codelets will not necessarily get a chance to put in their links, which is good, both because they are less natural and because we don't want the cytoplasm getting too cluttered up with piles of irrelevant relationships. Some examples of unlikely but potential connections of this sort are shown below:

How are such "unnatural" links suppressed? The suppression is carried out by musing codelets. In considering whether or not to insert a given link, they look at the density of links so far created, and once a certain "reasonable" density of links has been reached, they will (probabilistically speaking) let only the best potential actions proceed to be turned into action codelets and placed on the coderack. Let us assume that the musing codelets have done a good job of approval and disapproval, so that the cytoplasm is as notated in the diagram before this one.

At this point, the most syntactic or superficial processing has been done. We have "annotated" the cytoplasm's contents in syntactic terms, which will allow us to proceed toward more semantic terms. We are describing a rather ideal run, one in which pretty much everything goes well.

Moving into the Semantic Phase

Each time an adjacent S-link is created, this increases the *pressure* to search for S-groups. What this means in more detail is that for each S-link made, a musing codelet is created that will estimate the value of creating an S-group that contains the just-linked letters. As more adjacent S-links are made, more such musing codelets are made and placed on the coderack, and so the probability is ever rising that one of them will be run. When one eventually does run, it will probably approve of the S-group suggestion, and will launch an action codelet to actually carry out this action. Once it runs, the cytoplasm will contain its first genuine *semantic* structure (an S-group).

Such semantic structures will get created with a speed roughly proportional to the amount of lower-level (syntactic) activity agitating for them. It's a case of the squeaky wheel getting the oil, in a way:

lots of S-links add up to a loud clamor for seeking S-groups, and so eventually that pressure overcomes everything else, and such groups are indeed sought and found (or created, if you prefer to see it that way). This is quite typical of the way that several levels of indirect and statistical causation can be summarized metaphorically in terms of the intuitive concept of "pressure", and having spelled it out it once, we shall feel free to make use of the metaphor below.

In the case of our given analogy, most likely the S-groups ABC and XYZ would be recognized, and perhaps AB (inside ABD) would as well, although the credibility of a very short S-group is somewhat reduced, and so the musing codelet in charge of deciding whether or not to go ahead and instantiate it might veto the action. In fact, let us assume that ABC and XYZ are seen as S-groups, but that AB is not so recognized. (Or if it were, it might be marked as a somewhat "unhappy" S-group, in the same sense as structures in Jumbo could be unhappy.)

The Rule-Generation Phase

The crucial question now is: How does a *rule* get manufactured, describing the visible action in a plausibly intensional way? After all, this is where we have been headed all along: toward a reasonable perception not only of X1 and X1*, but also of their interconnections, so that finally a good description of the change will "fall out". We have sameness links from A to A and from B to B, and a successor-link from C to D. The question is, how are all these various pieces of information combined and abstracted into what will hopefully be one short, pithy rule?

We have a figure-ground dilemma here: in our rule, we could mention not only which piece or pieces changed, but also which pieces stayed the same. Alternatively, we could describe only what changed. The latter is more economical and more sensible. If you know what to change, the rest will automatically stay the same. This also saves you the effort of explicitly describing all the unchanged pieces, something that could get you into trouble. In ABC = = > ABD, the first *two* letters are unchanged -- but would you want to cast that in concrete? Isn't the essence simply that the rightmost letter *did* change? After all, with PQRS as target, it would seem bizarre if not absurd to preserve only the P and Q. So the solution to our figure-ground dilemma seems to be to describe only the *non*-sameness links.

In the present example, there is only one such link, and this focuses our attention on the roles played by the C and the D involved. This sounds like a very bland observation, but it exemplifies a deep and general truth: the effect of any change is to focus attention on some piece of the prototype, and to make us try to describe the *role* that that piece plays. The word "role" emphasizes to us that we don't want to think of the C in ABC as merely "one instance of Platonic C"; that would be crude and extensional, and it would blatantly ignore the salient fact that it is the *rightmost* element of ABC. How do we sense this saliency? What makes something salient, and how?

To each concept in the Slipnet there is attached a measure of its *semanticity*. This is a mere number, empirically assigned, which tries to estimate how useful the given concept is in making characterizations of an object. For example, letter types have low semanticity -- they produce overly extensional descriptions. Concepts such as "left-neighbor" and "right-neighbor" have slightly higher, but still fairly low, semanticity. Of higher semanticity are "predecessor" and "successor", and perhaps a bit higher are "rightmost" and "leftmost". Then of yet higher semanticity are "first" (which applies to the letter A alone) and "last" (applying only to Z), C-group, S-group, P-group, and others. Exact numerical semanticity values are yet to be assigned, and certainly changing them would dramatically affect the performance of the Copycat program. This will be a most interesting and undoubtedly critical experiment to perform, since different settings of these crucial numbers

surely result in very distinct and characteristic cognitive styles.

Although we have as yet no definitive table of semanticity, we do have some preliminary values and a mathematical way of combining them so that compound descriptions inherit semanticity from their components. No matter what scheme is finally adopted, though, the essential is that the semanticity value of a description should reflect both its *shortness* (high points for very short descriptions, of course) and the semanticities of its *components* (high-semanticity components would push the compound semanticity up). We expect that the semanticity values of various rival descriptions of C's role inside ABC will always come out in roughly this way:

	DESCRIPTION	SEMANTICITY RANGE
1.	"the third element of X1"	low
2.	"an instance of Platonic C"	low
3.	"two elements to the right of A"	low
4.	"the latest letter (in the sense of Platonic order) in X1"	medium
5.	"the other parameter of an S-group with parameter A (itself salient)"	medium
6.	"the rightmost letter of X1"	high

Description 5 sounds more forbidding than it really is. If ABC is recognized as an S-group, much of description 5 will be created instantly. The only part that will be missing will be the part that recognizes the salience of A itself.

Some of these descriptions will be attached to the node for that C token automatically. Number 2 will be present from the instant of the node's birth; number 6 will be present implicitly, in that there will be no right-neighbor link. If an enzyme for making adjacent S-links chances to try to look at that C's right-neighbor, it will find it nonexistent, and will so mark the C, which means that description 6 will at that time become explicitly a part of the node. Since this is an event of pretty high probability at a very early stage of processing, we will assume it has been done.

In any case, when it comes to making a rule describing the visible action, what we need most is intensional descriptions -- that is, *perceived roles* -- for the pieces involved. Luckily for us, the C under consideration does indeed have at least one strongly semantic (i.e., salient) role, and moreover it will very probably be attached to the node already (namely, number 6 above). Therefore, all we need now is a way of saying "replace ... by ...", where the ellipses symbolize intensional role-descriptions.

There are not too many wildly different typographical operations on groups of symbols, as Turing observed. All can be reduced to copying, erasing, inserting, and deleting. For our purposes, we would like to add a few natural categories that are just special ways of compounding the previously mentioned ones. For us, the primitives out of which all changes are composed are:

deletion (of one or several symbols)
insertion (of one or several symbols)
substitution (of one or several symbols for another)
exchanging two symbols (or groups)
reversal (of a group)
extraction (of one or several symbols)

Of course, this list is not sacrosanct. A few may be added, some changed or dropped. But for now, we have a repertoire of six templates into which intensional descriptions (i.e., role descriptions) need merely be plugged to obtain a complete characterization of most visible actions. (Sometimes more than one of these templates must be used, in which case they can be put together in a few standard ways.) In our case, clearly the action is substitution, and given that we have good intensional descriptions for C and for the relationship between C and D, we can say that the generated rule will be:

Substitute for the rightmost element its successor.

Of course, the rule will not be an English sentence, but a data structure made out of a template pointing to two intensional role-descriptions, but those implementation details do not concern us here. We should point out that other rules could easily be manufactured, since other descriptions of the C will probably exist, but their lower semanticities will not favor their use. Once again, the early bird gets the worm.

It would be useful at this point to contrast the likeliest rule to be suggested for "ABC = => ABD" with the likeliest one for "ABC = => C". That would be:

Extract the rightmost letter.

The fact that this is an extraction (and not, say, a substitution) is easily detected from the way the two compartments are linked up. Everything in X1* (namely, just the C) is derived from X1, and some pieces of X1 (A and B) are unlinked to X1*. Consequently, the best hypothesis is that one specific piece of X1 -- namely, the C -- has been extracted to form X1*.

The "Worlds-Mapping" and Rule-Slipping Phases

We have now reached the critical point where the two "worlds" -- the prototype ABC and the target XYZ -- are compared, to see if it makes sense to apply the given rule -- the "first-draft" rule -- *rigidly* or *flexibly*. Letter, or spirit? Well, of course, the spirit of Copycat is always to take the spirit of things, never the letter. On the other hand, sometimes there is no need to do any flexing, bending, or slipping; sometimes the letter will do very well. We have, after all, already performed a respectable degree of abstraction in our creation of the rule. That abstraction should make the rule apply in many situations. In fact, our alternate target PQRS points this up well. If you apply the rule to it, you get PQRT, just as reasonable people do -- not PQRD, PQSS, PQDS, PQD, PQRS, ABD, or any of the host of other bad but justifiable answers. So far, so good. The trouble is, if all Copycat can do is to look at the visible change, do a little bit of abstraction on it, and come up with a first-draft rule that is supposed to apply to any target equally well, then something is surely missing. It suffices to recall targets AAABBBCCC and RQP, for instance.

It is of the essence to be able to let the rule bend or slip freely, adjusting itself to the new target. Actually, that is not quite accurate. One cannot determine how the rule should slip by looking at the target alone. One needs to see how the target compares with the prototype. This, if anything, is the crux of the Copycat project:

How the rule should slip depends on how the "worlds" of prototype and target align; that is, how the internal *roles* inside them are perceived to match up. The two structures must be mapped as well as possible onto each other, with special account taken of any role (or roles) singled out in the first-draft rule. This mapping will create *pressures* on concepts utilized inside the rule. Only under such pressures will the rule slip and adapt itself to its

new domain, chameleon-like. Slippability under pressure is what turns a rigid rule into a pliable principle, a rigid analogy capacity into a fluid analogy capacity.

This is such an abstract statement that it must be illustrated by example. What happens when the target is AAABBBCCC or RQP? Should the rule be applied mindlessly? Some sort of check must first be done; otherwise you will rigidly turn AAABBBCCC into AAABBBCCD, and RQP into RQQ. And since you can't know in advance which targets will not require slipping, you must do such advance checking for all targets, even the simplest ones, such as PQR. The alignment of ABC and PQR appears simple: both are S-groups, both even have three elements. What could be easier? The (unchanged) first-draft rule is given the green light, runs, and PQS is produced. This is unproblematic.

Next target, RQP. Here, we have a P-group contrasting with the S-group ABC. Roughly speaking, our worlds align well, only backwards. How does the system discover this? The mapping of ABC and RQP is made relatively easy by the fact that both have high-level semantic descriptions: S-group and P-group, respectively. The fact that these top-level descriptors, both having high semanticity, are connected by a "symmetric-opposite" link in the Slipnet sends a powerful rush of activity into the "symmetric-opposite" *node* in the Slipnet. The principal effect of this higher activation is to put grease on all "symmetric-opposite" *links* in the Slipnet, which of course will accelerate any process wishing to slip the current rule into alternate versions of itself by means of "symmetric-opposite" links. In particular, the current climate now facilitates the replacement of "rightmost" by "leftmost". Thus the first-draft rule "Substitute for the *rightmost* letter its successor" is likely to slip into the following "translation", under the influence of RQP: "Substitute for the *leftmost* letter its successor".

Actually, one might object: Why didn't "successor" also slip into "predecessor"? Although such a slippage is indeed plausible here, there are considerable pressures militating against that double slippage, pressures that have to do with subtle features of the worlds-alignment. If P in RQP plays the role of A in ABC, then "leftmost" and "rightmost" have clearly been interchanged, but "earliest" and "latest" (referring to alphabetical order) have been preserved. Thus the exchange of symmetric-opposites seems to apply at a syntactic level only, not at the deeper level of "predecessor" and "successor". On the other hand, if R were mapped onto A, then "leftmost" and "rightmost" would be preserved while "earliest" and "latest" would be interchanged: pressure to do only *semantic* symmetric-opposite slippages. Thus the two best possible slippages of our first-draft rule, under the pressures created by target RQP, are:

(1) Substitute for the leftmost letter its successor;
(2) Substitute for the rightmost letter its predecessor.

The first is preferred, because it is based on a more deeply semantic view of the role of C in ABC: namely, it sees C as the *latest*, not just the *rightmost*, letter of ABC. Both of these rules, however, are quite plausible "translations" into the RQP-world of the first-draft rule.

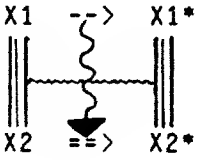
Rule Execution Phase

The system will now conclude by applying its new rule (whichever one it has created), and constructing one of the following analogies:

```
ABC ==> ABD          ABC ==> ABD
----------           ----------
RQP ==> SQP          RQP ==> RQO
```

Consider the first of these. SQP is the produced goal structure. To *close* the process, and to confirm its own sense of having made a good analogy, the system now needs to scan the new structure SQP, and to try to map it onto the given result ABD in the same way as RQP mapped onto ABC. It is of the essence that the same worlds-mapping should hold vertically on both sides of the arrow, and if so, that provides a clean closure to the analogy process.

Pictorially, it would look like this:



where the vertical connections symbolize world-mappings. The fact that the arrow from X2 to X2* is different from the other arrow symbolizes the fact that it comes from a "translation", rather than an exact copy, of the first-draft rule. The modulating influence of the worlds-mapping is symbolized by the vertical wiggly line connecting those arrows. The wiggly line crossing the diagram horizontally symbolizes the fact that the two world-mappings have been verified to be virtually the same.

Incidentally, had the target been AAABBBCCC, the C-group parameter values of A, B, and C would have mapped naturally onto the three letters A, B, and C in the prototype, and the thereby-greased link between "letter" and "C-group" would have facilitated the slippage of "rightmost *letter*" into "rightmost *C-group*". Thus, depending on the kinds of pressures applied to the rule, it slides in different and rather unpredictable ways into alternate versions of itself. (For related work on "prototype deformation" in legal reasoning, see **McCarty & Sridharan.**)

Hitting a Snag

Now let us finally tackle the case of target XYZ. We shall proceed in two steps. First we consider how XYZ would change given the visible action ABC = = > C; then we tackle the subtler visible action ABC = = > ABD.

We observed earlier that for the change ABC = = > C, the best rule would seem to be "Extract the rightmost letter", so let us assume that it has been created. The "next" phase is alignment or mapping of the two "worlds", ABC and XYZ. In some ways, these two worlds align quite easily -- ABC and XYZ are both S-groups, and they have the same length. An obvious one-to-one mapping suggests itself:

```
A   B   C
↕   ↕   ↕
X   Y   Z
```

We have not just a top-level mapping but even a complete role-to-role mapping, which is more than one could do for, say, prototype ABC and target PQRS. Since the mapping is so direct, no type of slippage whatsoever is called for. All systems seem go, and so we proceed with the extraction, coming up with the analogy

```
ABC ==> C
---------
XYZ ==> Z
```

This analogy is fine, and many humans -- probably most -- would see it as the best possible. This

might even encourage one to think that there is one single "natural" mapping of ABC and XYZ onto each other, and that -- whatever the rule might be -- you can feel confident that if it applies to ABC, it will carry over with no slippage to XYZ.

But the idea of "one mapping for all occasions" is a vain hope. ABC and XYZ are not identical; there is no unique or ultimate best way to map them onto each other. The fact that both are S-groups is certainly significant, and gives a strong go-ahead for trying a given rule without slippage; but it is not a guarantee. Different visible actions will involve different parts of the prototype, bringing out different roles and with different flavors. How these variable pressures will interact with a fixed target cannot be captured by a fixed mapping.

What warning signal might tell us in advance that we need to look for a new way of aligning the ABC and XYZ worlds other than our first, most naive mapping, shown above? Unfortunately, there will not necessarily be any warning. It may be necessary to run the naive rule, and simply see if it leads to a problematic result. This is of course precisely what happens when people try to apply to target XYZ the rule "Replace the last letter by its successor". They instantly run into a barrier: Z has no successor. And of course the system will encounter the same obstacle.

Fluid Recuperation from Snags via Slippability

This in effect forces the system to ask, "What kinds of slippage should be considered here?" It has the same effect on humans, of course. And as a remedy, many people are prone to suggesting seeing A as the successor to Z, and consequently they propose XYA as their answer. For them, the "obvious" slippage is not to be sought in a realignment of the two worlds, but simply in the very concept of "successor". "When Z's successor is asked for, just say A, even though you know it's not *really* Z's successor." This is actually a very creative slippage, and most likely its roots lie in analogies with other familiar linear orders that have some kind of circularity, such as the ten digits 0-9, the 13 cards in a suit (with ace both highest and lowest), the hours on a clock, days of the week, months of the year, and so on. However, Copycat does not have such experiences and knowledge to draw on, and as far as it knows, Z has no successor, period. To add machinery to Copycat that would allow it to extend Platonic concepts such as "successor" would be a most worthwhile but very difficult challenge; and then to come up with an analogy that would force Copycat to use that machinery (e.g., to see A as the successor of Z) would be a highly nontrivial problem.

Another thought is to slip the concept of "rightmost" in the most superficial way -- say, to "next-to-rightmost". This would convert the first-draft rule into "Substitute for the *next-to-rightmost* letter its successor", and the result would be XYZ = => XZZ. A similar thought would be, "If I can't take Z's successor, then why not take the 'nearest' thing, which is to say, Z's *predecessor*?" That would yield the answer XYZ = => XYY. Still another thought involves slippage in yet another direction: "If Z has no successor, then just give up -- drop the term completely." This yields XYZ = => XY.

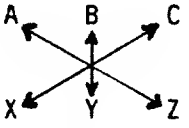
All of these thoughts are simple-minded patches applied somewhat arbitrarily, without any guidance. Admittedly, Copycat is supposed to be random at some level, but its randomness is at a level far lower than the level at which such serious decisions get made. We would hope that carefully controlled urgencies, many musing codelets, and the regularity of statistics would combine to weed out some of these more obvious but unmotivated slippages, and promote the re-examination of the ABC/XYZ mapping. Indeed, there are good reasons for doing so, completely aside from the failure of the rule, rigidly interpreted, to give a sensible answer.

There is strong pressure on Copycat to reexamine its mapping of XYZ and ABC. One source of such

pressure is localized: it is that the snag itself was local, and focused attention on the Z. The other source is general: during the parallel scanning of prototype and target, several concept nodes in the Slipnet became highly activated. A general rule of thumb is that a highly activated Slipnet is worth paying attention to. Given these sources of pressure, the system returns to the "worlds-mapping" phase and tries again, focusing specially on the Z.

The letter A in ABC activated the node for "first" (Platonically), while the letter Z in XYZ activated the node for "last". What is noteworthy about this is that "first" and "last" are nodes with a very high degree of semanticity, and moreover, they are connected by a "symmetric-opposite" link. Since they are such close neighbors in semantic space, they both contribute activation to each other and thus set up a strong self-reinforcing resonance. The combination of high semanticities with high activation levels means that codelets involving these concepts will be given very high urgencies.

In addition, A excited "leftmost" and Z excited "rightmost". These two nodes, although of lower semanticity than the corresponding Platonic concepts "first" and "left", are related to each other in the same manner -- i.e., via a "symmetric-opposite" link. Notice that the "symmetric-opposite" node has now been doubly activated. This puts grease on all the "symmetric-opposite" links. The high activation levels of the A and Z nodes, and the fact that the A and Z tokens occupy symmetric-opposite boundary positions in their respective structures (leftmost and rightmost) strongly suggests that the proper worlds-mapping involves not the simplistic alignment shown above, but a reversed alignment:



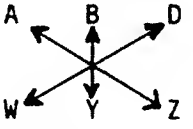
This is an example of "semantic symmetry" discussed above. Now, of course, XYZ's X clearly plays the role of ABC's C. And the same pressures as produce this worlds-mapping also have greased the proper links, thus making it very natural now for the first-draft rule to slip into:

Substitute for the leftmost term its predecessor.

With this new rule formulated, the system now goes ahead and applies it, producing as its answer WYZ, with no snags.

Closure-Checking Phase

Still, it will not settle down and decide that this answer is satisfactory until it has carried out its last self-confirming phase, that of evaluating the quality of the proposed answer. All four compartments of the cytoplasm are now filled, but the most recently filled one has no links of any sort in it. Therefore the syntactic and semantic scan phases are re-entered for this compartment, allowing internal links and semantic structures to be built up inside WYZ, and also allowing vertical links to be proposed between the two right-hand results (ABD and WYZ). There is pressure for A and Z to map onto each other, and of course for D and W to map onto each other as well.



This vertical correspondence of roles reveals the same semantic symmetry as does the worlds-mapping of ABC and XYZ. This fact thus happily closes the analogical circle, and sanctions the

proposed answer. (Had it not, we would have been sent back to earlier phases.)

DEEP ANALOGIES, PRESSURES, AND SCIENTIFIC REVOLUTIONS

We chose "ABC = => ABD; XYZ = => ?" as our representative analogy, because it well illustrates many desired features of the Copycat system. But another reason for choosing it is that it is such a fascinating case independently of the Copycat architecture. The contrast with "ABC = => C" shows that it is not at all obvious *a priori* that there are subtleties contained in the problem. They emerge only after one has tried out the simplistic rule and seen it fail. There is an initial "paradigm" by which one sees ABC and XYZ as nothing more than S-groups, and this overwhelms any other facts of the situation. Only when this view has brought about an insoluble problem does the other information, seething in the background, emerge into the foreground and take over. There is a sudden switch in what is considered relevant.

The parallel to the concept of a scientific revolution hits us over and over again. (See **Kuhn** and **Darden**.) How to structure one's perceptions of similarities? What is considered salient, and what incidental? What kinds of forces (mental events) are required to subtly alter the balance? What then ensues? What shifts, what gives, what emerges as dominant? What are the background aspects that remain stable during this process?

In our model of these questions, we view the Slipnet as the critical determiner of judgments. Not only the proximities that it defines, but also the levels of semanticity -- roughly determining a hierarchy of slippability -- combine together to define a total model of mental fluidity that allows analogical thought to emerge as a consequence of many interactions. It is this complex, statistically emergent view of mentality that we view as the most interesting potential outcome of our research. We feel certain that the hardest part of this work will be to control the stochastic processes, and to make them have a coherent focus most of the time. Probably the most important ideas to develop, once the basics of the architecture are in place and operational, involve the self-watching mechanisms, in which both the cytoplasm and coderack are monitored by the system, as sketched out earlier. There is no doubt that our work is cut out for us.

ACKNOWLEDGMENTS

Many people have contributed ideas to this work. Marek Lugowski has enthusiastically devoted a great deal of time and energy to the development of these ideas and this report, as well as supplying invaluable constructive criticism day and night. David Moser and I have had numerous conversations on the nature of analogy and on these analogies in particular. Paul Smolensky has contributed substantially to the development of my vision of mind as a statistical entity, and has given me impetus to defend my notion of an "ideal gas of analogy-ology". Gray Clossman has influenced me tremendously; he constantly comes up with new and original ways of looking at nearly everything. Marsha Meredith, in her work on Seek-Whence, has developed many ideas in parallel with me, and it has been exciting to bounce our ideas off each other and to watch them develop and slowly make more and more sense (we hope!).

Marvin Minsky invited me to MIT, where this report was written, and where some of the ideas were developed. I truly owe Marvin much more than a debt of gratitude, for he has done a great deal for my research efforts and for my determination to push these ideas forward. His grants have also supported me during my sabbatical-year stay at MIT. Many, many thanks are due to Marvin Minsky.

Other people who have made a significant difference in the working-out of these ideas or the writing of this report are: Robert Axelrod, Zamir Bavel, Donald Byrd, Daniel Dennett, Betty Dexter, Richard Feynman, Robert French, Dedre Gentner, Bernard Greenberg, Maria Moretti Gruenewald, Carl Hewitt, John Holland, David Israel, Pentti Kanerva, Scott Kim, David Leake, Henry Lieberman, Jay McClelland, Fanya Montalvo, Ryszard Michalski, John O'Donnell, David Rogers, Richard Salter, Vahe Sarkissian, Joe Shipman, Sue Stafford, Peter Suber, Jeremy Wertheimer, and Patrick Winston. I thank them all for their contributions.

APPENDIX 1

Connections to Other Work

We have attempted to indicate close connections to the ideas of others in the body of the report, but it is worthwhile listing some of them separately. The paper by Evans (Evans) described perhaps the first extensive AI work on analogy. That program solved IQ-test geometric analogy problems. Each problem was of the form "A is to B as C is to X", where for X one had a choice of five diagrams. Thus the program did not generate its answer, but merely picked one. While it was quite impressive in some ways, Evans' program succeeded largely because the variety of the problems was not all that great, and because the heuristics given to the program managed to cover many cases in the corpus he used. Few objects were involved in each diagram (thus little filtering for relevance was needed), and practically no higher-level relations were needed to perceive the answer. The control structure was completely deterministic, and proceeded through rigorous stages. The idea of levels of activation in a Platonic concept network was totally foreign to this work, as was the mixed bottom-up top-down strategy involved in the architecture of a perceptual system such as Hearsay II. We feel that an integration of these two ideas is required for work in understanding the nature of analogies to make real progress.

Two other early papers (**Becker** and **Kling**) dealt with analogy but once again in a deterministic framework without spreading activation. Kling's work especially, concerned with the transfer of proofs from one mathematical domain to another (e.g., groups to rings), stuck with analogies so perfect that "isomorphism" would be a better word for them. Indeed, category theory in mathematics is a mathematical equivalent of his work, revealing the exact commonality of structure behind certain classes of ideas. The difference is of course that true analogy is always imperfect and involves subtle balances and trade-offs, just as in perception.

There are a number of more recent (mostly current) projects we find close to ours in spirit, some of which are listed alphabetically below, with a few words of commentary:

ANDERSON: John Anderson's recent book *The Architecture of Cognition* (**Anderson**) presents his view of the centrality of spreading activation for controlling processes. Insofar as there is a resemblance between production system architecture and our probabilistic coderack architecture (a somewhat unclear mapping), there could be said to exist some abstract similarity of worldview here.

BACON: Behind the generic name "Bacon" are the names of several projects (including Bacon, Glauber, Dalton, and Stahl) and of several people (Herbert Simon, Patrick Langley, Jan Zytkow, and Gary Bradshaw). Their interest in scientific theory formation is consistent with ours. In Bacon, their focus has been on discovering a mathematical formula that encompasses all the data in a given set (vaguely related to the Seek-Whence project, although the resemblance is not much more than superficial), but in the later projects, they have been more concerned with the manipulation of concepts, putting them together in fluid ways to have maximum explanatory power. Potentially, this work has much in common with our work; however, analogies have so far not been given a high place in their work. (See **Langley et al** for brief descriptions of all these projects.)

DARDEN: Lindley Darden (**Darden**) has been working on understanding analogies from the point of view of a philosopher of science. Her work is only now drawing closer to

AI. We feel that our approach, based as it is on the concept of a "meta-domain", fills her objective of having AI people look for generalities across theories, i.e., *types* of theories. One of her principal interests is in understanding what differentiates good analogies from bad ones, and we feel that our work and Gentner's connect with her work in that way.

FELDMAN: The connectionist work by Feldman et al (see **Feldman et al**) at Rochester, concentrating on mostly low-level visual problems, bears a good deal of resemblance to work by Hinton et al at Carnegie-Mellon. The idea of "stable coalitions" of small computational elements relates to our view of regions in our concept network coming to an equilibrium of activation levels, with mutual reinforcement and mutual inhibition combining to make stability and to "lock in" certain perceptions. Very opposed to many traditional AI views, this work nevertheless "feels right" to us.

GENTNER: Dedre Gentner has been relentlessly pursuing the nature of good analogical thinking, and her work in "structure-mapping", and her stress on higher-order relations and systematicity parallel closely our approach stressing "semanticity". (See **Gentner**, for example.) In fact, her systematicity and our semanticity are very close concepts. Her group at BBN and our group at MIT are planning to have much contact in the coming few months.

HEARSAY II: This consists principally of Erman, Lesser, Reddy, and Hayes-Roth. The deep influence that Hearsay II had on this project is described in APPENDIX 3. (See especially **Reddy** and **Erman at al**.) Hearsay III (see **London et al**) is an extension of some of this work, and is less familiar to us, but is discussed somewhat in APPENDIX 4.

HOLLAND: John Holland's pioneering work on self-organizing systems that evolve intelligence through evolution (see **Holland:GEN**, for example) is close in spirit to our work, both in its dependence on stochastic processes and in its central tenet that "good" high-level structures can best be made from small pieces by hierarchical buildup combined with a randomized regrouping that largely respects chunks that have already been built.

KANERVA: Pentti Kanerva's work on memory (**Kanerva**) is a novel suggestion for how randomness can render memory search more efficient. Although his work is not directly related to ours, its stress on the importance of randomness for retrieving the "essence" of a situation is very consonant with our views, and colors our approach in some intangible but important ways.

LENAT: We feel Lenat's approach (see **Lenat:AM** and **Lenat:EUR**), although differing in implementation in many ways from ours, is motivated by many of the same questions. Both projects are fueled by intuitions about how discovery works in science and mathematics, as well as by interest in epistemology, the abstract nature of heuristics, and the relation between intelligence and evolution. Lenat's student Russell Greiner has written an interesting paper on analogies -- in essence, a compendium of analogies -- that reflects similar interests to our own.

LNR GROUP: From our perspective, this group includes, loosely, the following people: Donald Norman, David Rumelhart, Jay McClelland, Paul Smolensky, and, tangentially, Geoffrey Hinton. Norman and Rumelhart's work on simulation of a human typist (complete with errors), and Norman's work on explaining human errors, both based on their concept of activation levels of "schemas", as well as involving simulated parallelism, is very close indeed to our view of the "right" kind of architecture for a cognitive system. (See **Norman & Rumelhart** and **Norman**.) Our work differs from theirs, obviously, in its choice of problem (analogies) and domain (idealized), as well as in its stress on multiple levels of abstraction or semanticity. McClelland and Rumelhart's model of letter recognition in context (see **McClelland & Rumelhart**) not only shares many of the attractive architectural ideas of Norman and Rumelhart, but also involves multiple levels of structure (actually similar to Hearsay II). Furthermore, it shares our interest in perception of simplified "stick-figure" letter forms, although it does not come close to suggesting or treating the notion of stylistic analogy problems. Paul Smolensky (**Smolensky**) has integrated his notion of "computational temperature" into the McClelland-Rumelhart task domain, which further increases the cross-links between the LNR group's approach and our own. Finally, Geoffrey Hinton's work, with colleagues Sejnowski and Fahlman at Carnegie-Mellon University (see **Hinton & Sejnowski**), as well as in the book *Parallel Models of Associative Memory* (**Hinton & Anderson**), certainly overlaps with ours in that he sees cognition as emergent from statistical interactions of "subcognitive" elements (our codelets, for example), and he has been developing the notion of "computational temperature". This is discussed in more detail in APPENDIX 3.

MERLIN: This project was undertaken by James Moore and Allen Newell in the early 1970's (see **Moore & Newell**). Its premise was that a kind of conceptual mapping underlies all thought, which is entirely consonant with our view of analogy as the driving force behind cognition. They attempted to make a "recursive" analogy-maker, one in which a high-level mapping would "force" lower-level mappings until the process bottomed out in some already-known mappings. It was a fascinating idea, but never fully implemented, and it seems not to have been followed up.

MICHALSKI ET AL: Ryszard Michalski and colleagues have been working on many systems for learning, generalization, induction of patterns, and so on (see Chapters 4 and 11 in **Michalski et al**) . We feel a kinship to them not so much in terms of methodology espoused, for indeed theirs and ours are radically different, but because of a shared long-term intense concern with the role of abstraction and induction of patterns as the essence of intelligence.

TAXMAN: N. Sridharan and Thorne McCarty have for several years been developing a system that can carry out legal "reasoning". We put the word in quotes only to signify that it is not logic but analogic that enters here. In any case, their work on "prototype deformation" (see **McCarty & Sridharan**) is closely aligned with our entire model of analogy-making. Their strategy, on the other hand, is quite different to ours, being totally disconnected from randomness and perceptual architectures. Still, we have watched their work carefully, because the issues they are concerned with are so near ours. Tangentially related: Peter Suber, a philosopher and lawyer, has adopted some of our analogies as paradigms for

teaching the principles of legal reasoning in his courses on the philosophy of law (see **Suber**).

WINSTON: Winston (see **Winston**) is concerned with importance-weighted mappings of roles (slots) in one situation onto roles in another. His approach to determining "importance" is based on counting the number of constraints in which the given entity participates. We find ourselves largely in agreement with this aspect of his work. However, as he points out in the cited paper, in his work there are no clear principles about how levels of abstraction interfere with each other. Since this is one of our key foci (shared with Gentner), we find our work to be conceptually somewhat distant from Winston's work, especially insofar as his implementation employs a strictly top-down deterministic control structure. Our arguments against such a control structure are put forth by Hofstadter (see **Hofstadter:WHO**, **Hofstadter:SUB**).

YALE: The work on language and memory by the Schank group at Yale has always been concerned with deep issues of semantics, and gradually its focus has become the modeling of memory. Several people in this group have done very significant work on the structure of memory: Schank, Abelson, Kolodner, Dyer, DeJong, and Carbonell. We view their research as closely related to ours because in our pursuit of good analogy, we always must seek to skim off the irrelevant factors in a situation, and to find the core. "Core" is just a synonym for their semi-technical term "adage" (see **Dyer** particularly), and what interests us is how they get to the adage from the complex surface structure of a natural-language passage. Dyer's work involves a mixture of top-down and bottom-up processing, which has the perception-system bias that we favor. DeJong's recent work on "schema alteration" in story understanding (**DeJong**) involves some ideas related to our "worlds-mapping" phase. Kolodner's work on memory retrieval (see **Kolodner**, for instance) connects with our view of how activity spreads in the permanent concept network, although hers is focused on the problem of finding a memory whereas ours is concerned with perceptual control. Carbonell's work on metaphor (**Carbonell:MET**) involves what might be called "layers of slippability" (he calls it a "hierarchy of variability") and again parallels our concern with what slips most easily versus what is deepest in a given structure or situation. His other work on analogy (see **Carbonell:ANA**) reflects much commonality with us in its stress on modifying rules by transformations induced from mappings of contexts. Schank and Abelson's longtime collaborative work (most recently summarized in **Schank**) on the nature of memory structures interests us because of its concern with highly abstract analogies (their code word for that is "reminding"). These are the kinds of analogies that have most inspired us, as well, and we are influenced by all examples and analyses of how this might be explained. The Schank book's MOP's and TOP's, etc., are closely related to Dyer's structures, and have some parallel in how we organize our concept network and tokens in our cytoplasm.

APPENDIX 2

Analogies, Domains, Frameworks, and Roles

People understand each other's situations in life largely by projection. A woman can understand what it means to a man that his wife has died of cancer, despite the fact that she is not a man, that she is not married, that she has never known anyone who has died of cancer. She may bring to bear a number of sad circumstances in her own experience -- things as remote, on a superficial level, as a loss of job or having her house burgled. In fact, she will probably rely on a vast number of her past experiences, all balanced unconsciously, in her attempt to project herself into his situation vicariously.

True understanding is made of this kind of thing. It involves looking for resemblances at the deepest possible level, and being able to ignore shallow resemblances unless they also go deeper. For each candidate situation retrieved from memory, it involves carefully testing its "isomorphism" to the matter at hand. Clearly, "isomorphism" is far too stringent a term, but the essential thing is for the two mapped situations to share some internal structure: for each to have salient internal *roles* and for there to be a comfortable correspondence between some of the major roles. How does one locate the roles -- major and minor -- in a situation? How does one find which roles in a totally different situation map onto the given situation's major roles? What if there are conflicts? What if everything works smoothly except for some small hitches? What if there are two completely different ways of carrying out the mapping, each with its own advantages and disadvantages? This is the essence of the human dilemma: no two things are *ever* the same, and yet we must make do with finite minds and finite category systems to understand, as best we can, the complex world we are embedded within. We must always simplify and yet try to preserve the *essence* of what we perceive. And that is what the study of analogies must focus on.

Consider the ability to transfer a skill from one domain to another -- say, learning to drive a motorcycle given that one can already drive a car. Here, one has to adapt to a new type of clutch, new ways of shifting gears, and so on. Yet in a deeper sense, the essence is the same. This ability to adapt to a new situation by recognizing its shared essence with an old situation is the crux of learning. It seems to amount to being able to tell how features -- we call them "roles" -- of the new domain are counterparts of familiar roles in the old domain, and to overlook minor discrepancies between the ways the roles are instantiated in the two cases. These two abilities are interdependent, for clearly, one has to be able to overlook minor discrepancies in order to spot major similarities that establish roles as each other's counterparts; and yet conversely, ignoring minor discrepancies is far easier once one has identified the roles inside a new situation and knows which roles they should map onto in a familiar old situation. Learning, like understanding, is thus a typical analogy problem.

One area of life where making analogies is explicitly recognized as an art and as an objective way of getting at the truth is law. There, every decision is a judgment based on resemblance of the current case to one or more precedent cases. (See **Suber**.) The way a mapping is carried out is of the essence. What is to be ignored in the current case, and what focused on? In terms of what concepts is the current case best framed? In terms of what concepts have precedent cases been framed? Can an alternate set of concepts be chosen, and new or old cases be recast more convincingly in terms of them? Can a more abstract set of concepts be brought to bear, to make a higher-level analogy between the current case and some other precedent, superficially more distant but abstractly more similar?

The Copycat Domain: Both Micro-World and "Meta-Domain"

Analogies can be good or bad, compelling or weak, deep or shallow, elegant or ugly, and so on. We are interested in a model that, in a given situation, may come up with one, two, or even several analogies, but we demand that any analogy produced should be appealing and perhaps even elegant, rather than simply defensible in some weak or silly way. In other words, we want our model not ever to entertain bad analogies. By "not entertain", we do not mean that it might internally produce bad analogies but then censor them from public view; we mean that bad analogies should literally not be produced at any stage. We also require that the system be able to explain its analogies -- not in a natural language, but simply by exhibiting all the ingredients that go into them: the roles perceived, the correspondences perceived, the degrees of semanticity, and so forth.

In this project, we are studying analogies -- especially elegant ones -- in a domain that is an abstraction of many domains. Our domain is small, yet rich and strikingly subtle. The ideas should not, therefore, be taken entirely at their surface level, as if we thought that this domain per se were an important one. The ideas are about how analogies work *in general*, and we feel this domain is an ideal one for examining, in microscopic detail, analogies and the pressures that make them work or not work.

A word of explanation is needed here -- in fact, many words are needed, but we shall keep it brief. A reader might easily make the following type of objection: "Interesting analogies usually do not connect concepts *within* a single domain; what makes them interesting and gives them power is that they *cross* domains. Therefore, it seems misguided to study analogies in a single domain, especially right after having stated that the essence is in making unusual, stunning, surprising -- but compelling -- jumps." We reply as follows. Over several years, we have tried to isolate the essence of this "jump-making" capacity, and we have found that the essence does *not* lie in the fact of differing domains per se; rather, it seems to lie in the ability to find highly abstract and unanticipated connections -- let us say, unpreprogrammed similarities -- between structures. Put another way, the essence of leap-making seems to involve the ability to jump between different descriptions of structures, shifting boundary lines, levels of chunking, degrees of abstraction, making associations between neighboring concepts in a semantic space, and so on.

With all this in mind, we have slowly distilled our one domain from observations about cross-domain analogies. We have carefully tailored it so that within it, we can capture the essence of creative, spontaneous, unanticipated analogies, reduced into a miniature universe. We have made a vast number of analogies in this domain that require each one of the above types of ability, sometimes requiring several at once. We shall show and explore a few sample analogies in what follows. Making a system that can solve them as humans do is our goal.

The ability to elicit unanticipated abstract connections is, we feel, the key ability needed for spontaneous analogy generation, whether within a single domain or crossing domains. Thus our domain, despite its small size, symbolizes the universe of *all* concepts and categories, in which cross-domain analogies exist. As such our micro-world is a "meta-domain".

APPENDIX 3

Connections with Recent Temperature Work and with Hearsay II

The departures from strict iterative improvement (one-way traffic that can lead to stagnation) may seem reminiscent of recent work (see **Kirkpatrick et al**) on optimization by simulated annealing, in which a system of many independent variables tries to minimize an "energy" function or cost function (in essence providing a measure of the total "unhappiness" of the current state) by trying out small local changes that affect the energy value, always accepting them if they reduce the energy, and probabilistically accepting them even if they increase it, where the probability of accepting an energy-increasing change goes down exponentially with the amount by which the energy increases, according to the formula

$$p = e^{-E/T}$$

E being the (positive) change in energy, p being the probability of accepting a change of size E, and T being a so-called "computational temperature" -- an adjustable parameter. This Boltzmann formula, first suggested in Monte Carlo calculations by Metropolis, then adopted by Kirkpatrick et al in their work on optimizing combinatorial searches, and more recently adopted by Smolensky (see **Smolensky**), as well as Hinton and Sejnowski (see **Hinton & Sejnowski**), for use in creating a new type of relaxation algorithm for AI systems, introduces a precise mathematical notion of computational temperature, which amounts to a variable controlling how much risk one is willing to take in backing away from a solution in which one has already invested some time. The larger T is, the easier it will be to take an "uphill" (i.e., energy-increasing) step, while the closer T is to zero, the more unlikely it is that an uphill step of a given size will be taken. The principle of annealing is to start out with a high temperature, allowing all sorts of random steps to be taken, and slowly to cool the system down by lowering the value of T, while many random steps are proposed and possibly taken. Of course, all downhill ones will be taken, while uphill ones may or may not. The idea is that a nonzero temperature provides a kind of "jiggling" that keeps the system from getting stuck in local energy minima -- and a slow, controlled descent to zero temperature allows the system to explore various pathways before settling down in one final state, presumably a globally happy one.

The resemblance to the Jumbo system is quite close. In fact, the notion of "system temperature" was introduced -- and by that name -- in Jumbo approximately a year before the paper of Kirkpatrick et al appeared. Jumbo's system temperature is derived from the happinesses of the top-level gloms in the cytoplasm. It is highest when the top-level word candidate is least happy, which is to say, most in need of an escape route from what is apparently an unfavorable region of the search space. Such an escape route would be provided by the introduction into the coderack of breaking enzymes. The temperature is lowest -- approaching freezing, or zero, when the top-level word candidate is deemed quite satisfactory by all criteria known to the system, so that there is no need to seek any radical alternative solutions.

Jumbo's musing codelets correspond, to some extent, to the various randomly proposed steps in the annealing process, of which some are accepted and some are rejected. Both involve the notion of making small forays into a counterfactual world and "testing the waters", so to speak, before deciding whether or not to actually *risk* taking such a step. The difference is mainly that in the annealing model, the decision whether to take such a step is made mathematically, on the basis of the

Boltzmann formula, while in Jumbo, the decision is more spread out, and depends on how the various musing codelets evaluate the prospect of the given change, and this in turn depends on the notion of happiness, which is related to the system temperature. Another complexity is that the speed with which a chain of musing codelets gets run depends on the urgencies of its constituent codelets, which in turn are governed by the judgments of previous codelets in the same chain. This means that considerably more levels are involved in Jumbo's acceptance or rejection of a cytoplasm-changing action than are involved in an annealing model's acceptance or rejection of an energy-changing step. Nonetheless, the resemblance is more than superficial, particularly in that both types of system see a need for a variable regulating the amount of disorder to be deliberately injected into the system to render its search for a good state more efficient, and both types of system begin with a high temperature and move towards a "freezing" state, monitoring their progress along the way in order to determine how to control the temperature.

In both types of system, any overall state that is finally settled upon by the system has necessarily been reached through a series of slow adjustments during a period of gradually increasing order and decreasing temperature, mediated by the cooperative action of many small operations, some moving towards and some moving away from the top-level goal. In Jumbo, the decrease in temperature is not necessarily monotonic, since the self-monitoring of the system can result in a decision to raise the temperature, but certainly the temperature does fall to zero during any successful run. Thus, our system has an extra degree of flexibility of allowing uphill steps in temperature, which in effect means that the system is annealing at the meta-level as well. In both systems, many parts of the space are "sniffed at" without actually getting visited, and in both it is to be hoped that the final solution achieved represents a close-to-optimal global state, which is necessarily a compromise, in that it must reconcile a large number of mutually incompatible local desires (the technical term for this is "frustration").

Some of the strategy of Jumbo was inspired by the Hearsay II speech-recognition system of Carnegie-Mellon University (see **Reddy** and **Erman at al**). The parallel exploration of various parts of the space to different levels of depth -- something referred to in Jumbo as the "parallel terraced scan" -- was inspired by, though it differs from, some of the methods used in Hearsay for activating knowledge sources through demons with various levels of "conditions", "preconditions", and "pre-preconditions". The musing codelets, in particular, correspond (roughly) to these tests performed before a knowledge source is let run. The actual running of a Blackboard-modifying knowledge source corresponds to the running of a cytoplasm-modifying codelet -- an "action codelet".

Jumbo's cytoplasm, as was already mentioned, resembles Hearsay's Blackboard in some ways. In particular, both are the loci where all domain data structures are stored and modified. Moreover, they both contain different representations of the lowest-level data (the raw waveform in Hearsay, the set of elementary letters in Jumbo) at various levels of abstraction, or "chunkedness". One noticeable difference is that in Jumbo, there is no "third dimension" -- the dimension reserved in Hearsay for several alternative hypotheses involving the same parts of the waveform. In Jumbo, this would correspond to a single letter belonging to two or more disjoint gloms. In Jumbo, such shared structure simply does not exist. There are two reasons for this. One is that we believe that this brand of parallelism simply does not exist in the mind: we cannot really hold in mind at one and the same time several rival structures composed of the same lower-level units. A model based on allowing rival hypotheses to coexist in its global data structure must suffer the consequences when rival hypotheses arise for different pieces of the whole -- the number of rival *compound* hypotheses is of course the product of the numbers of rival *simple* hypotheses. This quickly produces a combinatorial explosion of hypotheses, something to be strongly avoided. A second reason for avoiding this type of parallelism is that it violates our cellular metaphor. In particular, it would be analogous to a single

atom being in incompatible states or belonging to disjoint molecules at one time, which is of course impossible. If we wish to keep the cellular metaphor and remain faithful to our own vision of the mind's activities, these two reasons militate against the third dimension of the Blackboard, or cytoplasm..

A final difference between Jumbo and Hearsay is the way in which Jumbo is permeated by randomness. Hearsay attempted to make every single decision about scheduling, focus of attention, knowledge source activation, and so on, *intelligently*. This seems to be attempting to use too much power, too much intelligence, when often there is little basis for such decisions. The Jumbo philosophy is to try to do your best by assigning to every codelet an urgency, and to split up the work of evaluation of various potential pathways into rival pathways of musing codelets. The control structure of Jumbo is thus much simpler and more fluid, since it does not need to worry about what to do next: that is always determined by the luck of the draw. Jumbo profits from statistics, however, in that over a period of time, luck cancels out and the best pathways will tend to prevail. However, there is always the chance that quirky pathways will be explored as well (in fact, there is a *certainty* that such pathways will be at least sniffed down, by musing codelets). Thus, every pathway actually taken is surrounded by a "halo" of non-taken but sniffed-at potential pathways. The importance of the wide swath of nearly-taken pathways through the space of possibilities becomes most evident in the projects for which Jumbo was always seen as a mere prelude, such as Seek-Whence, Letter Spirit, and Copycat.

APPENDIX 4

Self-Monitoring in its Various Guises

The kind of self-monitoring system we describe at the end of the section on the architecture of Jumbo is somewhat reminiscent of Hearsay III (see **London et al**) and of the work on 3-Lisp and Mantiq by Brian Smith (see **Smith**). The principal difference between our approach and that of Hearsay III, with its extra blackboard devoted to the scheduling of processes, is that we are relying on the musing codelets to choose good pathways in advance, and on self-watching techniques to correct for musing codelets' mistakes *a posteriori*, whereas the Hearsay III effort is trying to use maximal intelligence *a priori*. A similar distinction in style can be drawn between Jumbo and 3-Lisp. Smith's interpreter that watches itself at all levels *during* its operation is very different from ours, in which the system watches only the *results* of its operations, and thus is shielded from fine-grained knowledge of how they came about. The system does not debug or alter itself at a fine-grained level, since it has no access to itself or knowledge about its own workings, but simply makes sure that it significantly alters some of its biases and then tries again. We believe that this more closely resembles genuine cognition with its trial-and-error quality. People cannot look inside their heads and modify their brain processes at a fine-grained level.

APPENDIX 5

How Analogies Blur the Line Between Extensionality and Intensionality

If the basic change were ABC = => ABQ instead of ABC = => ABD, we could not get any link made between the C and the Q; they are simply unrelated letters. On the other hand, what if the change were ABC = => ABE? C and E are not exactly distant from each other in the alphabet, and yet there is no direct "hard-wired" Platonic link between C and E in the Slipnet. Does this mean that C and E are virtually as far apart as C and Q, to Copycat? Not at all. When the C node in the Slipnet is activated, its activation tends to spread to the D node, since it is an immediate neighbor. Likewise, the E tends to excite the D node. These two shoves push the D node's activation level way up, enough that it can be used as a mediator between the C and E tokens in the cytoplasm. Indirectly, then, their connection will have been noticed. On the other hand, a more remote alphabetic connection, such as between C and F, will be considerably less likely to be picked up on, because more Platonic mediators are needed. And for C and Q, of course, the situation is hopeless. This has deep repercussions, by the way. The visible action ABC = => ABQ will be understood only extensionally -- either as "C changes to Q" or "rightmost letter changes to Q". A visible action such as ABC = => ABE will be construable both extensionally and intensionally, and ABC = => ABD nearly always intensionally. Thus there is a kind of fuzzy borderline between extensionality and intensionality that emerges quite naturally from the program's architecture. This is one of the subtlest features of human cognition, because it has everything to do with the way we project one situation onto another.

If Mary tells Ann, "My brother died", and if Ann does not know Mary's brother -- that is, there is no extension, for Ann -- then how can she understand this statement? Surely projection is of the essence: Ann will imagine her *own* brother dying (if she has one -- and if not, then her sister, or a good friend). This intensional view of the situation allows Ann to empathize with Mary. Even if Ann knew Mary's brother a bit, she might flicker between thinking of him as the person she vaguely remembers and thinking of her own brother dying. Here is an example of the same sort of blurry mix of extensional and intensional understandings of one and the same situation. This kind of blur is typical in many situations, and accounts for some of the introspective opacity that analogies seem to have for most people. (See **Hofstadter:SHK** and **Kripke** for ideas on intensionality, identity, and Al.)

REFERENCES

(Anderson) John R. Anderson, *The Architecture of Cognition*. Harvard University Press, 1983.

(Becker) J. Becker, "The Modeling of Simple Analogic and Inductive Processes in a Semantic Memory System", *Proceedings of the First International Joint Conference on Artificial Intelligence*, Washington, D.C., 1969.

(Bongard) M. Bongard, *Pattern Recognition*, Hayden Book Co. (Spartan Books), 1970.

(Carbonell:ANA) J. Carbonell, "A Computational Model of Analogical Problem Solving", *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, August, 1981.

(Carbonell:MET) J. Carbonell, "Metaphor -- A Key to Extensible Semantic Analysis", *Proceedings of the 18th Meeting of the Association for Computing Linguistics*, 1980.

(Darden) L. Darden, "Reasoning by Analogy in Scientific Theory Construction", *Proceedings of the International Machine Learning Workshop*, Monticello, Ill., June 1983.

(DeJong) G. DeJong. "An Approach to Learning from Observation", *Proceedings of the International Machine Learning Workshop*, Monticello, Ill., June 1983.

(Dyer) M. Dyer, *In-Depth Understanding*, MIT Press, 1983.

(Erman at al) L. Erman, F. Hayes-Roth, V.Lesser, and D.R. Reddy, "The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty", *ACM Computing Surveys*, Vol. 12, No. 2, June 1980.

(Evans) T. Evans, "A Program for the Solution of Geometric-Analogy Intelligence Test Questions", in M. Minsky, *Semantic Information Processing*, MIT Press, 1968.

(Feldman et al) J. Feldman and D. Ballard, "Connectionist Models and Their Properties", *Cognitive Science*, Vol. 6, No. 3, 1982.

(Gentner) D. Gentner. "Structure-Mapping: A Theoretical Framework for Analogy", *Cognitive Science*, Vol. 7, No. 2, 1983.

(Hinton & Anderson) G. Hinton and James A. Anderson, eds., *Parallel Models of Associative Memory*, LEA, 1981.

(Hinton & Sejnowski) G. Hinton and T. Sejnowski, "Optimal Perceptual Inference", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Washington, D.C., June 1983.

(Hofstadter:GEB) D. Hofstadter, *Gödel, Escher, Bach: an Eternal Golden Braid*, Basic Books, 1979.

(Hofstadter:JUM) D. Hofstadter, "The Architecture of Jumbo", *Proceedings of the International Machine Learning Workshop*, Monticello, Ill., June 1983.

(Hofstadter:MMM)
D. Hofstadter, "Metafont, Metamathematics, and Metaphysics: Comments on Donald Knuth's Article 'The Concept of a Meta-Font'", *Visible Language*, Vol. 16, No. 4, Autumn, 1982. Also: Indiana University Computer Science Department Technical Report No. 136, December 1982.

(Hofstadter:MT1) D. Hofstadter, "Metamagical Themas: Roles and Analogies in Human and Machine Thought", *Scientific American*, Vol. 245, No. 3, September 1981.

(Hofstadter:MT2) Hofstadter, D., "Metamagical Themas: On Sphexishness, 'Jootsing', and Creativity", *Scientific American*, Vol. 247, No. 3, September 1982.

(Hofstadter:MT3) D. Hofstadter, "Metamagical Themas: On Variations on a Theme, Slippability, and Creativity", *Scientific American*, Vol. 247, No. 4, October 1982.

(Hofstadter:OSW)
D. Hofstadter, "On Seeking Whence", forthcoming Indiana University Computer Science Department Technical Report, 1984.

(Hofstadter:SHK) D. Hofstadter, G. Clossman, and M. Meredith, "Shakespeare's Plays Weren't Written by Him, But by Someone Else of the Same Name: An Essay on Intensionality and Frame-Based Knowledge Representation Systems", Indiana University Computer Science Department Technical Report No. 96, July 1980.

(Hofstadter:SUB) D. Hofstadter, "Artificial Intelligence: Subcognition as Computation", Indiana University Computer Science Department Technical Report No. 132, November 1982.

(Hofstadter:WHO)
D. Hofstadter, "Who Shoves Whom Around Inside the Careenium?, or, What is the Meaning of the Word 'I'?", *Synthese*, Vol. 53, 1982. Also: Indiana University Computer Science Technical Report No. 130, 1982.

(Holland:ARM) J. Holland, *Adaptation in Natural and Artificial Systems*, Chapter 5, "The Optimal Allocation of Trials", University of Michigan, 1975.

(Holland:GEN) J. Holland, "Learning and Classifier Systems", in *Machine Learning II*, Michalski, Carbonell, and Mitchell, eds., Tioga Press, forthcoming.

(Kanerva) P. Kanerva, "Self-Propagating Search: A Unified Theory of Memory", Ph.D. dissertation, Stanford University 1984.

(Kirkpatrick et al)
S. Kirkpatrick, C. Gelatt, Jr., and M. Vecchi, "Optimization by Simulated Annealing", *Science*, Vol. 220, No. 4598, 13 May 1983.

(Kling) R. Kling, "A Paradigm for Reasoning by Analogy", *Artificial Intelligence Journal*, Vol. 2, pp. 147-148, 1971.

(Kolodner) J. Kolodner, "Retrieval and Organizational Strategies in Conceptual Memory: A Computer Model", Ph.D. dissertation, Yale University, 1980.

(Kripke) S. Kripke, *Naming and Necessity*, Harvard University Press, 1980.

(Kuhn) T. Kuhn, *The Structure of Scientific Revolutions*, University of Chicago Press, 1962.

(Lakatos) I. Lakatos, *Proofs and Refutations*, Cambridge University Press, 1976.

(Langley et al) P. Langley, J. Zytkow, H. Simon, and G. Bradshaw, "Mechanisms for Qualitative and Quantitative Discovery", *Proceedings of the International Machine Learning Workshop*, Monticello, Ill., June 1983.

(Lenat:AM) D. Lenat, "Towards a Theory of Heuristics", in *Methods of Heuristics*, R. Groner, M. Groner, and W. Bischof, eds., LEA, 1983.

(Lenat:EUR) D. Lenat, "The Nature of Heuristics III: Eurisko: A Program that Learns New Heuristics and Domain Concepts: Program Design and Results", *Artificial Intelligence Journal*, Vol. 21, March 1983.

(London et al) P. London, L. Erman, and S. Fickas, "The Design and an Example Use of Hearsay III", *Proceedings of the 7th IJCAI*, Vancouver, B.C., 1981.

(McCarty & Sridharan) T. McCarty and N. Sridharan, "The Representation of an Evolving System of Legal Concepts II. Prototypes and Deformations", *Proceedings of the 7th IJCAI*, Vancouver, B.C., 1981.

(McClelland & Rumelhart) J. McClelland and D. Rumelhart, "An Interactive Activation Model of Context Effects in Letter Perception: Part 1. An Account of Basic Findings", *Psychological Review*, Vol. 88, No. 5, September 1981.

(Michalski et al) R. Michalski, J. Carbonell, and T. Mitchell, *Machine Learning*, Tioga Press, 1983.

(Moore & Newell) J. Moore and A. Newell, "How Can Merlin Understand?", in *Knowledge and Cognition*, L. Gregg, ed., LEA, 1974.

(Norman) D. Norman, "Categorization of Action Slips", *Psychological Review*, Vol. 88, No. 1, January 1981.

(Norman & Rumelhart) D. Norman and D. Rumelhart, "Simulating a Skilled Typist: A Study of Skilled Cognitive-Motor Performance", *Cognitive Science*, Vol. 6, No. 1, January-March 1982.

(Reddy) D. Raj Reddy, et al, "Working Papers in Speech Recognition, IV: The HEARSAY II System", Carnegie-Mellon University Computer Science Department Technical Report, February 1976.

(Restle) F. Restle, "Structural Ambiguity in Serial Pattern Learning", *Cognitive Psychology*, Vol. 8, 1976.

(Schank) R. Schank, *Dynamic Memory: A Theory of Reminding and Learning in Computers and People*, Cambridge University Press, 1982.

(Smith) B. Smith, "Reflection and Semantics in a Procedural Language", Ph.D dissertation, Massachusetts Institute of Technology, 1982.

(Smolensky) P. Smolensky, "Schema Selection and Stochastic Inference in Modular Environments", *Proceedings of AAAI-83*, Washington, D. C.

(Steiner) G. Steiner, *After Babel*, Oxford University Press, 1975.

(Suber) P. Suber, "Analogy Exercises for Teaching Legal Reasoning", unpublished paper, Earlham College, Richmond, Indiana.

(Winston) P. Winston, "Learning by Augmenting Rules and Accumulating Censors", Proceedings of the International Machine Learning Workshop, Monticello, Ill., June 1983.